"DESIGN OF TELEROBOTIC SYSTEM OVER A LOCAL AREA NETWORK (LAN)"

BY

ABDULKHALIQ J. ALHARTHI

A Thesis Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

SYSTEMS ENGINEERING
January, 2002

Al-Harthi, Abdulkhaliq Jraib Salem

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by *Abdulkhaliq Jraib Salem Alharthi*

under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

Thesis Committee

Thesis Advisor (Dr. Onur Toker)

Co-Advisor (Dr. M. Al-Muhammed)

Member (Dr. Fouad M. Al-Sunni)

Department Chairman

Dean of Graduate Studies

8/1/2002

Date

Dedicated to

My Parents, My Brother,
My Sister, My Wife,
and My Children:
Kholoud, Abdulrahman, and Saud

Acknowledgment

First and foremost, all praise to Almighty Allah who gave me the patience to carry out this work successfully.

Acknowledgment is due to the King Fahd University of Petroleum and Minerals for supporting this research.

I wish to express my deep appreciation to my thesis advisor Dr. Onur Toker for his guidance and continuous support. I would also like to describe my gratitude towards my thesis co-advisor Professor Mayez Al-Muhammad for his constant help and countless hours he devoted for this work. Thanks also are due my committee member Dr. Fouad Al-Sunni for his interest and cooperation.

My thanks also to the chairman, faculty and my friends, who made my stay at KFUPM enjoyable and unforgettable one.

Contents

Acknowledgment iii

List of Figures vii

Abstract (English) xi

Abstract (Arabic) xii

1 INTRODUCTION 1

1.1 Statement of the Problems 8

1.2 Organization of the Work 8

2 LITERATURE REVIEW 10

2.1 Control of Remote Manipulator 12

2.2 Communication Delay 14

2.3 Visual Display 15

3 ROBOT MOTION COORDINATION 17

3.1 Robot Arm 17
3.1.1 Robot Structure 17
3.1.2 The PUMA-560 Manipulator Arm 19
3.2 The Motion Coordination Concept 19
3.2.1 Cartesian Frames 23
3.3 Direct Geometric Model of PUMA-560 28
3.4 The Geometric Model for Master Arm 40

4 VISION BASED MASTER ARM 59
4.1 RGB Color Space 62
4.2 Vision Program 63
4.2.1 User Interface 64
4.3 Part I: Detection and Tracking 66
4.4 Part II: Three Dimensional Computations 82
4.4.1 Affine Invariant from Multiple Views 83
4.5 Part III: Experimental Human-Robot Interface System 86

5 TELEROBOTICS 88
5.1 The Client-Server System 89
5.1.1 The Client Environment 89
5.1.2 The Server Environment 92
5.1.3 The Client-Server Interaction 97

5.2 Teleoperation with Master Arm 99

5.3 Mappings 106

5.3.1 The First Mapping Scheme 106

5.3.2 The Second Mapping Scheme 109

5.3.3 Mode Selection 114

6 EXPERIMENTS AND PERFORMANCE ANALYSIS 118

6.1 Vision System Performance 118

6.1.1 Single Camera Performance 119

6.1.2 Three Dimensional Computations Performance 122

6.2 Telerobotic System Performance 127

6.2.1 Experiment1: Teleoperation without Video Cameras 130

6.2.2 Experiment2: Teleoperation with Two Video Cameras 130

6.3 Actual Experiments 136

7 CONCLUSIONS AND RECOMMENDATIONS 138

7.1 Conclusions 138

7.2 Recommendations 140

Bibliography 141

List of Figures

1.1 Telerobotic system for repairing high-voltage power lines in Japan. . . 4

3.1 A prismatic joint versus a revolute joint 18

3.2 Robot arm: joint rotation 20

3.3 Robot arm: member identification 21

3.4 Frame R_1 rotated and translated relative to R_0 22

3.5 Tool frame relative to Base frame 23

3.6 Mappings between kinematic descriptions. 24

3.7 Two links with a revolute joint 27

3.8 Reference arm position 30

3.9 Frame representation for the PUMA-560 arm 31

3.10 The master arm 40

3.11 Frame representation for the master arm 42

3.12 O_2^* relative to O_2 45

3.13 O_3^* relative to O_3 47

3.14 The new structure of the master arm 52

3.15 O_6 relative to O_3^* 54

4.1 The 3D object 60

4.2 The primary colors: R,G, and B 63

4.3 User Interface 65

4.4 Vision program flow chart 67

4.5 The supervisory teaching mode in the vision program 68

4.6 The red ball used in the experiments 70

4.7 RGB components for the red ball 70

4.8 The MVD function for the red ball with white background at both sides 72

4.9 The MDD function for the red ball with white background at both sides 73

4.10 The HD function for the red ball with white background at both sides 75

4.11 Search area, boundaries, and diameter 79

4.12 The coordinate system for the 3D object 83

4.13 Experimental system 87

5.1 The Client-Server system 90

5.2 The Client-PC Interface 93

5.3 The Server-PC Interface 96

5.4 The Load/Initialize step 98

5.5 The Active system state 100

5.6 The final shape of the master arm 102

5.7 The first end-effector design 103

5.8 The effector part with a small angle for joint 5 104

5.9 The modified effector part 105

5.10 The final design of the effector part 105

5.11 The first mapping scheme. 108

5.12 The second mapping scheme. 112

5.13 The mathematical interface. 114

5.14 The PUMA-560 frames. 115

6.1 Relative error in X direction 120

6.2 Relative error in Y direction 120

6.3 Straight line trajectory at moderate speed 123

6.4 Straight line trajectory at high speed 123

6.5 Single Circular trajectory 124

6.6 Multi Circular trajectory 124

6.7 Single line trajectory in the 3D space 126

6.8 Multi line trajectory in the 3D space 126

6.9 Single circle trajectory in the 3D space 128

6.10 Multi circle trajectory in the 3D space 128

6.11 Timing in the telerobotic system 131

6.12 Contribution of each value in the system 131

6.13 Latency measured by the client and the server 132

6.14 Contribution of the client and server as complete system 132

6.15 The effect of the stops between motions on the client 133

6.16 The effect of the stops between motions on the whole system 133

6.17 Timing in the telerobotic system with video streams feedback 134

6.18 Latency measured by the client and the server 135

6.19 Comparison between the average timing 135

6.20 Stacking objects 136

6.21 Writing Allah 137

THESIS ABSTRACT

Name:	Abdulkhaliq Jraib Salem Alharthi
Title Of Study:	Design of Telerobotic System over a Local Area Network (LAN)
Major Field:	Systems Engineering
Date of Degree:	January, 2002

In this thesis, a telerobotic system has been designed, to allow remote control of a robot arm existing in the college of computer science and engineering laboratories, over the college network and by using a master arm that has been built locally. The system is developed for situation or environment that are too dangerous or uncomfortable for the human operator to perform, to keep him away and in a safe place. The operator receives real time video streams from the robot side, and accordingly he performs his work. Vision based master arm, based on un-calibrated stereovision to compute the three-dimensional information of a geometric shape from multiple views, have also been proposed and implemented.

Master of Science Degree

King Fahd University of Petroleum and Minerals

Dhahran, Saudi Arabia

January, 2002

ملخص الرسالة

الاســـــــــــــــم: عبدالخالق جريب سالم الحارثي

عنوان الرسالة: تصميم نظام للتحكم بالأذرعة الآلية عن بعد من خلال الشبكات المحلية

التخصـــــــــــص: هندسة النظم

تاريخ التخـرج: يناير ٢٠٠٢م

في هذه الرسالة، تم تطوير نظام متكامل للتحكم عن بعد بذراع الية موجودة في مختبرات كلية علوم وهندسة الحاسب الالي، وذلك عبرالشبكة المحلية الموجودة في الكلية و بواسطة ذراع تحكم تم تصميمها محليا. وقد طور النظام للأماكن الخطرة أو التي يصعب التواجد فيها لكي يتمكن المشغل من العمل في أجواء امنة وبعيدة عن المكان الذي توجد فيه ظروف غير مناسبة للتواجد. وفي هذا النظام يتم تزويدالمشغل بصور تلفزيونية مباشرة من موقع الذراع الالية يمكنه العمل من خلالها. كما تم اقتراح وتطوير ذراع تحكم غير تقليدية تعتمد على استخلاص معلومات وقياسات الأبعاد الثلاثة لشكل هندسي من خلال كاميرات فيديو غير معايرة.

درجة الماجستير في العلوم

جامعة الملك فهد للبترول والمعادن

الظهران- المملكة العربية السعودية

شوال ١٤٢٢ه

Chapter 1

INTRODUCTION

A robot is an automatic apparatus or device that performs functions ordinarily ascribed to human beings, or operates with what appears to be almost human intelligence, while, robotics is the science and art of designing and using robots. The machine that extends a person's sensing and/or manipulating capability to a location remote from that person is called a teleoperator. The term teleoperation refers most commonly to direct and continuous human control of the teleoperator [43]. A telerobot is an advanced form of teleoperator which can be defined as a remote control of manipulators or vehicles by a human operator with supervisory and some direct control [43]. However, the design issues for robots and telerobots are essentially the same.

The term telepresence means that the operator receives sufficient information about the teleoperator and the task environment, displayed in a sufficiently natural way,

that the operator feels physically present at the remote site. One of the key features in telepresence is to provide the viewer with the capability of observing 3D scene remotely as if real [48].

Telerobotics devices are typically developed for situation or environment that there are too dangerous, uncomfortable, limiting, repetitive, or costly for human to perform. Some applications for telerobotics are listed below [43]:

Space: assembly, maintenance, exploration, manufacturing, and science. The primary current example of a space teleoperator is the 20-meters-long remote manipulator system (RMS) built by the Canadian firm SPAR and carried aboard US space shuttle. It has six degrees of freedom and is controlled directly by a human operator viewing through a window or over video and using two three-axis variable rate-command joysticks. New telerobotic roving vehicles and manipulators are being designed for landings on the moon and on mars and other planets.

Process control plants: nuclear, chemical,.. etc, involving operation, maintenance, emergency. The nuclear "hot laboratories" continue as the application most wedded to force-reflecting master-slave systems. Vertut and his CEA laboratory near Paris have developed what has probably been the most extensive and successful set of such systems. Teleoperators have been valuable in the cleanup of the nuclear power station accidents at Three Mile Island and Chernobyl.

Underwater: inspection, maintenance, exploration, science, and surveying. For example, by 1980 "remotely operated vehicles" (ROVs) had come into extensive use by

the offshore oil and gas industry in well head completion operations, monitoring of pipelines, placing of sacrificial anodes and inspection of welds on subsea structures and other tasks.

Military: operations in the air, undersea, and on land. Examples of these are inspection and deactivation of mines and other explosives, observation of enemy operations without exposure to enemy fire, and remotely piloted air craft.

Medical: telediagnosis, telesurgery, and remotete treatment. One form of teleoperator is the endoscope, a coherent fiber-optic bundle with tubes for conveying fluids to or from the distal end point, wires for modifying its curvature so it can be maneuvered around corners, and additional wires operating an end-point gripper or snare or cauterizer.

A recent example of an industrial teleoperation system is shown in Figure 1.1. Generally, telerobotics are concerned with the following functionalities [33]:

1. Interfacing master and slave workstations so that the operator see the slave arm and feel the force feedback that are sent from slave work station through the network. Using the above information the operator can remotely manipulate the slave robot arm to achieve some tasks.

2. Providing a library of intelligent operator services to support the operator, for example:

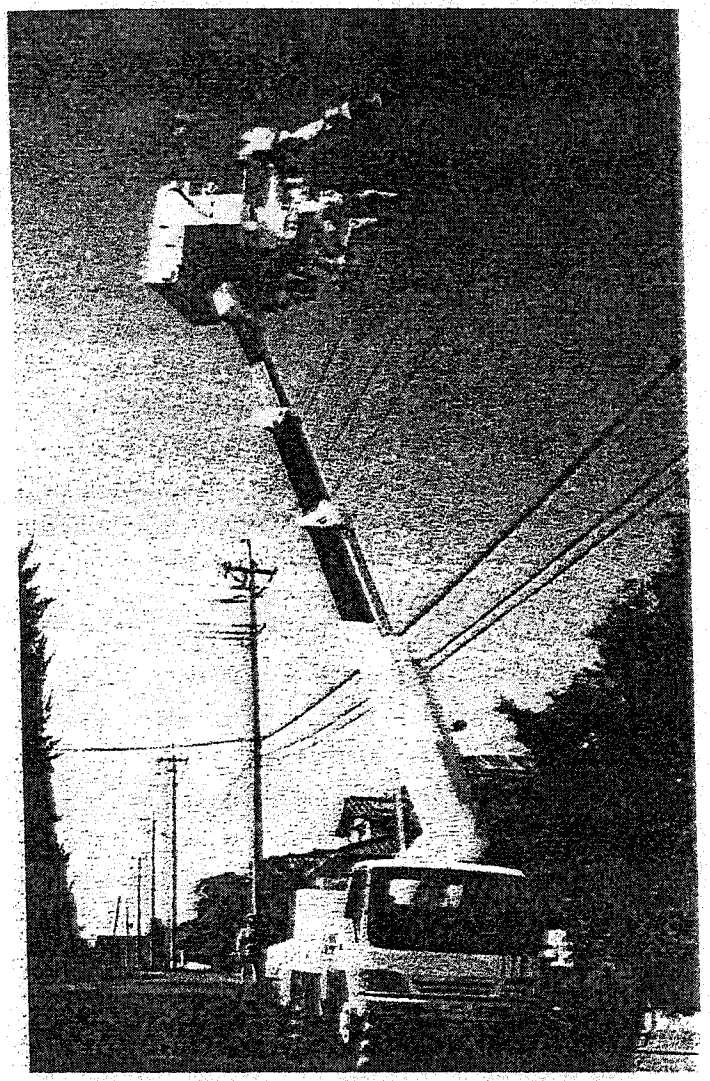

Figure 1.1: Telerobotic system for repairing high-voltage power lines in Japan.

(a) Ability to remote control of orientation zooming of cameras in slave system

(b) Ability to remotely activate some local loops for slave arm using sensors.

(c) Ability to shift the master arm from a non-dexterous configuration to another without causing changes to the position of slave arm

(d) Ability of operator to interact in real time with workstation through network

3. Studying the problem of time delay and delay jitter caused by network latency and their effects on telerobotics. Evaluate the performance of networked telerobotics and its operability in the presence of network delays.

Most traditional telerobots have closed loop control with vision feedback, force feedback or both and the operator closing the loop. In master-slave systems the operator applies international forces via a joystick or control force utilizing sensory feedback such as visual feedback and reflected force on the master arm for evaluating the current control state against his intention. Control of the remote manipulator may be via direct human operator control, autonomous robot control, or a combination of the two. In direct control the human operator need good concentration. If the communication delay is very large then supervisory control can be introduced where the remote device operates in largely autonomous mode. With the supervisory con-

trol the operator communicates a goal to the telerobot then the goal is executed by a local control system through artificial effectors and sensors and supervised by the operator.

Visual feedback plays an important role in the fields of object tracking, assembly, inspection tasks and many other applications. The visual information could be the guide to the planning and the teaching of manipulator's trajectory. Therefore, achieving an adequate image of workspace and transmitting it to an operator is essential to a successful accomplishment of tasks. There have been a lot of researches in the area of telemonitoring systems. The structure of telemonitoring systems can be classified into two types according to their mobility. One has a movable camera system that is able to modify its position and orientation according to the change of the environment, the other system has a fixed camera structure in the workspace, where the camera cannot change its configuration. To overcome this drawback, multiple cameras are often used. Reality enhanced teleoperation environment where the operator can feel as if he is in a remote place is known as tele-existence or telepresence. One of the most important elements of tele-existence is the visual display, which present the 3D world to the viewers. Most of the conventional 3D displays give the depth of the image by presenting different images to the right eye and the left eye respectively. For example, 3D displays with stereoscopic glasses, HMDs, and most of the 3D display systems without stereoscopic glasses adopt this mechanism [19].

Internet-based control systems must rely on the available communication protocols to exchange real time data between the controller and the process. Today, most network protocols provide a transparent and reliable support for data exchange among computers by using the Transmission Control Protocol (TCP). This protocol provides a full duplex stream service, with automatic error handling, retransmission, packet re-ordering, and guarantee of safe delivery. However, from the point of view of a real-time application, this protocol has the drawback of having unpredictable arrival time of the data. Researches on the communication time delay problem can be grouped into one of the following three categories [27]:

1. Teleprogramming: Supplying high-level commands to the robotic planner. The goal is to imbue the machine with enough intelligence to carry out the abstract task independently.

2. Predictive displays: Using predictive visual feedback modules to compensate for the communication delay. Often, a live (though delayed) video feed is also sent back from the remote site. This is to give the human teleoperator a greater sense of "telepresence" in the remote environment.

3. Bilateral control: Force information is returned from the remote environment. A force reflective device or video output of forces is used to immerse the teleoperator's senses in the remote environment.

1.1 Statement of the Problems

Design of a real telerobotic system application over the KFUPM local area network involves the following objectives:

1. Solving the geometric model of the robot and the master arms.

2. Developing an intelligent application program that allows the operator to control and manipulate the slave arm remotely. This step incorporates some tasks, such as developing a client-server system and interactive user interface.

3. Solving the problem of the difference in the geometric design between the master and the slave arms, which will be called mapping between arms.

4. The last part of the project has focused on building a vision based master arm. This part deals with the vision problems and the use of vision in this research.

1.2 Organization of the Work

Chapter 1 offers a brief review of the teleoperation and telerobotic systems. Chapter 2 provides a brief history and literature review of the field of teleoperation. Chapter 3 deals with the robot motion coordination and geometric modeling of the robot arms. Chapter 4 deals with vision and extraction information from images and using these information to control the robot arm. Chapter 5 is divided into two parts: the first part presents the client sever system, while the second part deals

with the development of mapping schemes between the master and the robot. Performance analysis and actual experiments are presented in Chapter 6. Conclusions and recommendations are provided in Chapter 7.

Chapter 2

LITERATURE REVIEW

Telerobotics systems date back to the need for handling radioactive materials in 1940's. Around 1945 the first modern master-slave teleoperators were developed by Raymond Goertz at Argonne National Laboratory, near Chicago. R Geortz 1950's developed mechanical teleoperator by which radioactive materials in a "hot cell" could be manipulated by an operator outside the cell. At that time the interconnection was entirely mechanical (steel, cable, wire, and ribbon) and therefore the distance between the master and slave station was limited to few meters. His mechanical devices eventually evolved into systems using electrical servos and cameras (Goertz and Thompson 1954) [14]. From the early 1960s telemanipulators and video cameras were being attached to submarines by the US, USSR, and French navies and used experimentally. The early lunar teleoperator called Surveyor demonstrated vividly the "move-and-wait" problems of time delay in an actual space mission. Soon

thereafter supervisory control was shown to offer a way around the time-delay problem (Ferrel and Sheridan 1967). By the 1970 the western interest in teleoperation had turned to undersea applications, for there was great economic demand for offshore oil. French developed their ERIC vehicle, the Americans the Hydroproducts RCV150. These were small unmanned submarines with remotely controlled video and manipulation capability plus the necessary thrusters for maneuvering. Since 1970 the pace of teleoperator development has quickened. The principal developments in teleoperation have been for undersea application and for space application [43].

The first live camera accessible through the World Wide Web was setup by researchers at Cambridge university to monitor status of coffeepot at www.cl.com.ac.uk/ coffee/ coffee.html. In September 1994 an ASEA IRb-6 robot was connected to the Internet through a web server at university of western Australia by ken Taylor [26]. Just few weeks earlier ken Goldberg at the university of California, Berkley had connected a SACRA type robot to the Internet [25]. These were the first physical devices connected to the web. In an ambitious project, the German aerospace corporation successfully launched ROTEX aboard the space shuttle Columbia in 1994, ROTEX was a robotic arm which could perform simple construction operations based on a ground-based operator's directions or an autonomous controller. In April 1997 Kevin Brady pulled a joystick in Albuquerque that controlled the motions of PUMA robot in Tarn's laboratory more than 1500km away, the experiment

takes 3 minutes involved avoiding box in its path to perform a manufacturing task of picking up an object and place it somewhere else, this experiment provides another dimension to the Internet real time application [49]. By 1997, the number of robots connected to the web was 5.

Telerobotics research area involve problems in control of remote manipulator, communication delay, and visualization.

2.1 Control of Remote Manipulator

Because of inadequate in purely manual or purely autonomous approaches, the semi-autonomous approach introduced to allow obstacle avoidance in teleoperated system, where commands from an input device are superimposed onto autonomous motion allowing human intervention to avoid obstacles as in [15]. The bilateral feedback is one of the most effective schemes for the Hi-Fi teleoperation, with which the force is feedback to a master from a slave [24]. The term bilateral refers to two-way information flow between the control input device (master arm) and the remote manipulator (slave arm). In this mode of control there is a feed forward position loop, which makes the slave arm follow the master arm movement and a force feedback loop from the slave arm to the master arm which makes the operator senses all forces encountered during manipulation [2]. A distributed teleoperation systems over the Internet was presented in [5, 54]. In these systems any number of robots, agents,

and operators connected to the Internet can communicate and interact together to achieve remote tasks. NASA's pathfinder mission exploited this. Initially, scientists had to travel to the control center in California to program the robot on Mars. An Internet interface was developed and it became possible for scientists to collaborate, and control the pathfinder mission from any where in the world. One attempt to combine human and machine control was made by Hayati and Venkataraman [18]. In their attempt, force and velocity commands from a master input device were combined with those from an automatic controller along each direction to be controlled. Bakes [4] presented a controller that superimposed various preprogrammed motions onto the command from the master, which could be initiated when desired. An event-based controller, which provided more automatic selection of controller parameters, was used by Guo et al. [15] to allow semiautonomous obstacle avoidance in a teleoperated system. In this simple implementation, velocity commands from an input device are superimposed onto an autonomous motion, allowing human intervention to avoid obstacles. In [9] a method of variable mapping of master to slave motion was applied to manipulation assistance using processed images. The master velocities were amplified or attenuated based on the commanded motion, the desired configuration of the slave, and the relative position of the slave based on processed end effector camera images. The variable velocity mapping was used to assist in the approach and grasping of a cross-shaped object. The first extensive application of force-reflection (FR) for the ground teleoperation of a space robot was

presented in [40] . It has been demonstrated that despite time delay, there are ways in which FR can be successfully used to improve the performance of the operator. In contrast to the standard force-reflecting master-slave systems, a general-purpose force-reflecting hand controller (FRHC) has been implemented at Jet Propulsion Laboratory (JPL). The hand controller is a six-degree of freedom control input device that can be back driven by forces and torque sensed at the base of the end effectors of a remote robot arm. This hand is general purpose, and can be coupled to any remote slave arm through appropriate mathematical transformation implemented in a computer control system. The computer-based control system of the FRHC supports four modes of manual control: position, rate, force-reflecting, and compliant control in task space (Cartesian space) coordinates.

2.2 Communication Delay

For networks with large time delay, researchers have developed many kinds of software tools. For example, Sensor feedback control or strategy has been developed in many systems for adjusting position and orientation of a slave arm at a remote site. For telerobotics applications where the time delay is greater than one second and it is varying, the instability problem can be eliminated by the supervisory control. Predictive display of a remote site has been developed in a 3-D graphics computer at a master site in [53] to overcome the problem of time delay. In [35]

a shared autonomy is designed under the sensor-based motion-planning algorithm where the position-feedback and force feedback are synchronously processed under the impedance control. In [17] a teleprogramming control paradigm for most telerobotic applications is presented which can enhance the performance of a teleoperator system in the presence of significant communication delay. Soda [46], presented a method for model based bilateral control of a master-slave arm with time delay between master and slave. In the proposed system the operator can manipulate the master arm based on signals from the simulator model to control the slave arm without confusion of time delay. In [12] a method based on H_∞ optimal control and μ-synthesis framework is introduced to design a controller for the Teleoperator that achieves stability for a pre-specified time-delay margin while optimizing performance specifications. Igor [21], developed an efficient system for remote robot control over WWW using JAVA 3D virtual robot representation with a local path planner, as opposed to commonly used Video images. This approach overcome the problem of delayed video images and provides suitable control condition for the operator.

2.3 Visual Display

In [13] multiple cameras was used and a method for selecting the optimal image among multiple ones and transferring it to the operator was presented. Although many studies show advantages of stereoscopic vision over monoscopic vision, [16]

shows that the design of a user interface for Australia's telerobot on the Web had to make the use of the available monoscopic views. It gives an example of how visual enhancement can compensate most of the disadvantages of multiple monoscopic views.

The Jet Propulsion Laboratory (JPL) advanced teleoperator (ATOP) project developed a method for a Hi-Fi calibration of graphics image to actual image. First they create Hi-Fi 3-D graphics model of robot arm, then overlay this graphics over the actual 2-D video images, the motion control drives the real robot as well as the graphics, thus the operator can see the consequences motion commands in real time, before sending the commands to the remotely located robot [1]. This technique has been demonstrated under several seconds communication delay on a large laboratory scale in may 1993, involving the JPL as the simulated ground control station and 2500 miles away, the Goddard Space Flight Center as the simulated satellite servicing set-up [53].

Chapter 3

ROBOT MOTION COORDINATION

3.1 Robot Arm

3.1.1 Robot Structure

An industrial robot is a multi function manipulator that can be modeled as an open chain of rigid bodies, called *links*, connected in series by kinematic joints. The function of the joint is to control the motion between the links. The first link is attached to the supporting base by the first joint, and the last link contains the end effector or other type of manipulator device. Each *joint-link* pair constitutes one degree of freedom. An n degree of freedom manipulator contains n joints, or in more

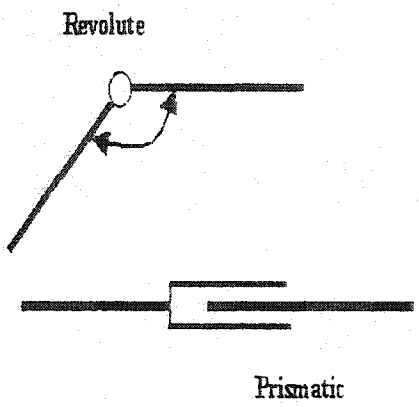

Figure 3.1: A prismatic joint versus a revolute joint

general terms, n link-attached coordinate system. The joints and links are numbered starting from the base. The lowest joint is fixed to the reference coordinate system, while the highest joint is fixed to the local coordinate system of the end effector. Robotic joints can be categorized as either *revolute* or *prismatic* joints as shown in Figure 3.1. A revolute joint allows link L_{i+1} to rotate with respect to the previous link L_i. A rotation angle θ_{i+1} can be used to define the angular position of L_{i+1} relative to L_i. A prismatic joint allows a link to translate with respect to the previous link. A translation variable θ_{i+1} can also be used to define the linear position of L_{i+1} relative to L_i. The notation used in this chapter is taken from [33] "Introduction to Robotics" by M. Al-Muhammed. Our project is implemented on the PUMA-560 robot arm which will be explained in more detail in the next Section.

3.1.2 The PUMA-560 Manipulator Arm

The PUMA-560 robot arm has six degrees of freedom and all are rotational joints. The last three have concurrent rotation axes, which simplify their geometric and kinematic models. The joint axes and range of rotation are shown in Figure 3.2. All the joints are driven and controlled by DC-Servomotors. The servomotors are equipped with electromechanical brakes that can lock the arm in a fixed position. The brakes are released by the controller when the arm power is on. The components of the robot arm are the *Trunk*, *Shoulder*, *Upper Arm*, *Forearm*, *Wrist* and *Mechanical interface* as shown in Figure 3.3. Functionally this arm can be divided into two parts which are the "transporter" and the "effector" parts. The transporter is responsible of transferring and positioning the effector which include the grasping system and the work piece. On the other hand, the effector is responsible for the orientation of the arm. The transporter includes three links that are the shoulder, elbow and forearm, while the end effector part includes the *pitch*, *yaw* and *roll*.

3.2 The Motion Coordination Concept

In the study of robotics one studies the location of objects in the 3-dimensional space. The objects are the links of the manipulator, and the tools with which it deals. These objects are described by just two attributes: their position and their orientation. In order to describe the position and orientation of an object in space,

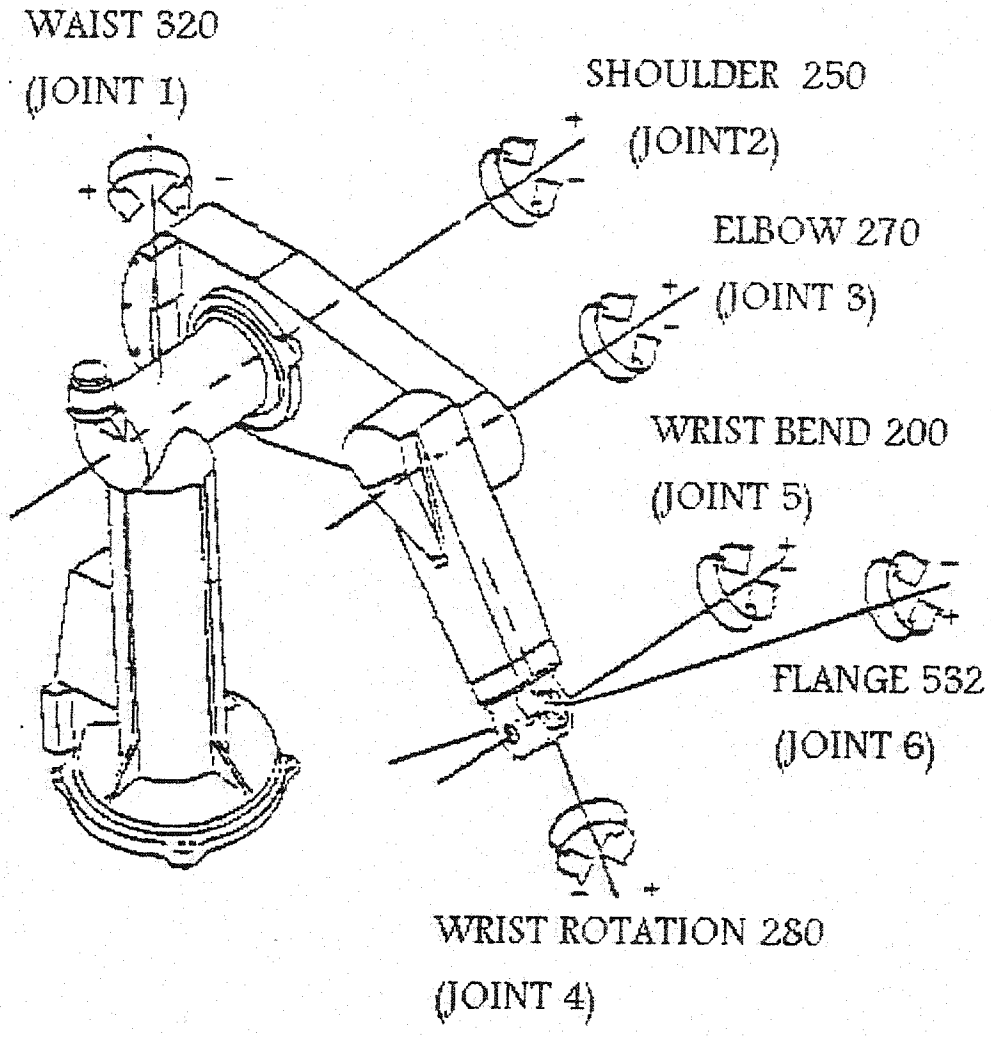

Figure 3.2: Robot arm: joint rotation

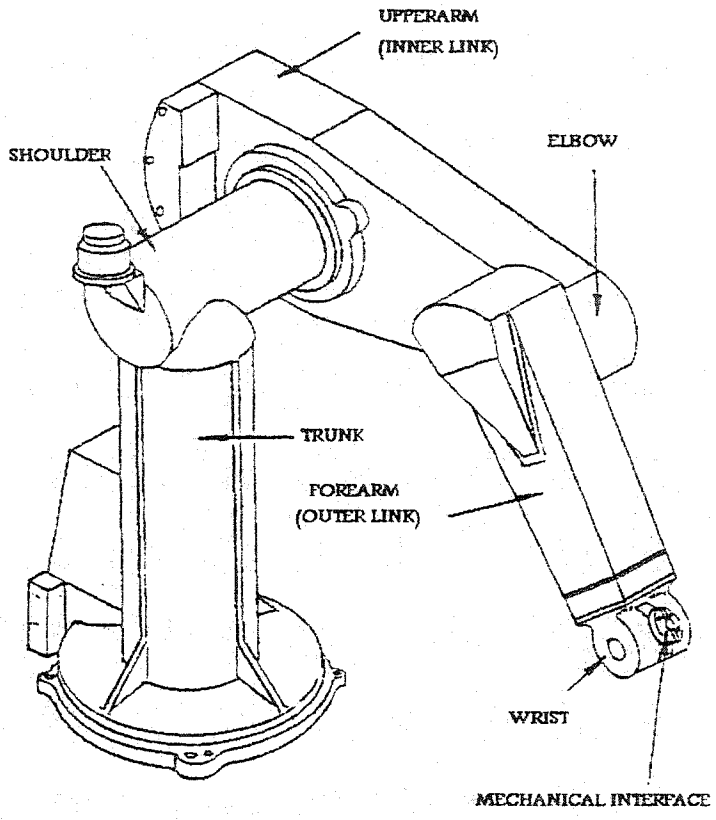

Figure 3.3: Robot arm: member identification

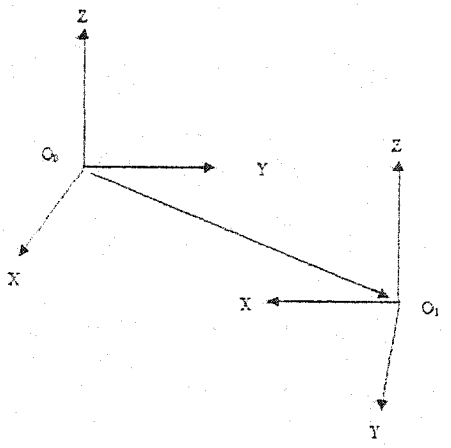

Figure 3.4: Frame R_1 rotated and translated relative to R_0

we will attach a frame of reference to each link. The frame of reference is defined using three orthonormal vectors $\{X, Y, Z\}$. Figure 3.4 gives an example of frame R_1 translation and rotation relative to frame R_0. The position of the manipulator is generally described by giving a description of the tool frame, which is attached to the end effector, relative to the base frame which is attached to the fixed base of the manipulator as shown in Figure 3.5. The relative position and movement of the individual links with respect to their preceding links provide a description of an entire articulated structure in the operating space and formulate a mathematical model of a kinematic chain of the robotic system. Since robotic manipulation can be achieved only by maneuvering the arm linkages in the task's environmental space, robot kinematics is an important tool in work space design, trajectory planning and motion rate control. Kinematics is concerned with the analytical description of spatial position, orientation, displacement, velocity and acceleration. There are

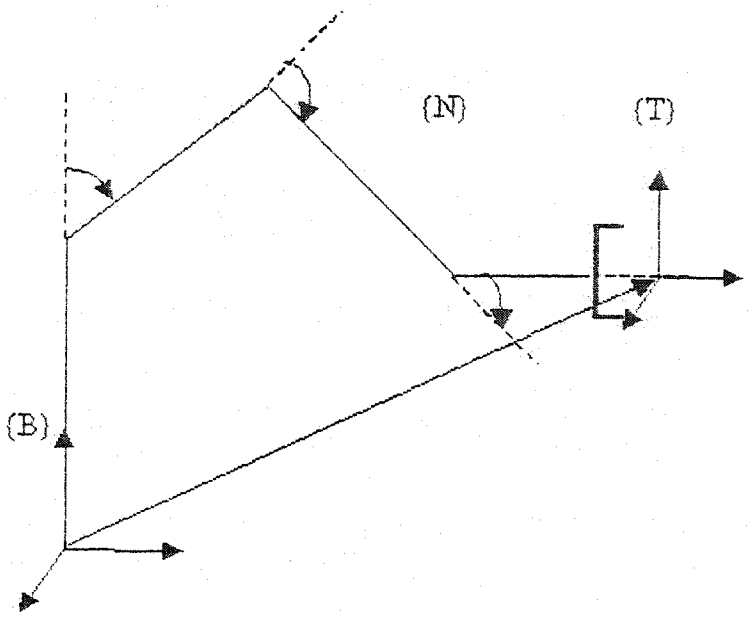

Figure 3.5: Tool frame relative to Base frame

two fundamental problems in studying robotic kinematics: the direct kinematic and the inverse kinematic problem. The direct kinematic problem involves the determination of the position and orientation of the end effector with respect to the reference coordinate system, given the joint variables of the robot arm. The inverse kinematic problem, on the other hand, involves the determination of the joint controlled variables, given the position and orientation of the end effector.

3.2.1 Cartesian Frames

In the 2-dimensional space there are three degrees of freedom (DOF): X, Y and θ orientation parameter. In the 3-dimensional space there are six DOF ,three position parameters: X, Y, Z and three angular orientation parameters specifying the orien-

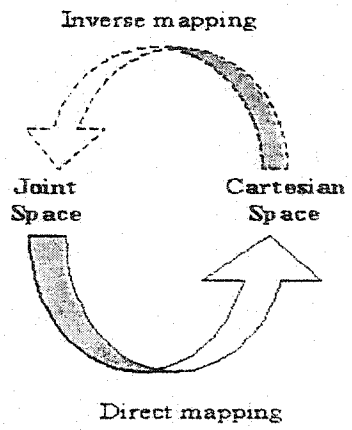

Figure 3.6: Mappings between kinematic descriptions.

tation of the gripper in the space. We specify both position and orientation using a translation vector (3x1) and a rotation (orientation) matrix (3x3) as suggested in [33]. These notations are opposed to the Denavit-Hartenberg (DH) notations which are used in the majority of cases and represented by a 4 x 4 homogeneous matrix that transforms a vector from one coordinate system to another. A Cartesian coordinate system is defined in the three dimensional space, by introducing three orthonormal vectors X, Y, Z. A frame can be defined by the orthonormal vectors with origin O. We say that the link L_{i+1} revolute with respect to link L_i when frame R_{i+1} rotates relative to either axes X_i, Y_i or Z_i as shown in Figure 3.7. The end point O_{i+1} of L_{i+1} can be associated a vector O_iO_{i+1} which will be denoted by $O_iO_{i+1,i}$ to indicate that the vector is observed in frame R_i. $M_i^{i+1} = [X_{i+1,i}, Y_{i+1,i}, Z_{i+1,i}]$ is the transfer matrix from frame R_{i+1} to R_i, which represents the rotation between links L_i and

L_{i+1}. Therefore, the link vector $O_iO_{i+1,i}$ can be expressed as follows:

$$O_iO_{i+1,i} = M_i^{i+1}.O_iO_{i+1,i+1} \tag{3.1}$$

where $O_iO_{i+1,i+1}$ denote the vector O_iO_{i+1} observed in frame R_{i+1}. Note that the vector $O_iO_{i+1,i+1}$ has simple expression bcause link L_{i+1} is parallel to axis Z_{i+1}. Therefore, the position vector $O_0O_{n,0}$ can be decomposed as sum of the link vectors:

$$O_0O_{n,0} = \sum_{i=1}^{n} M_0^i.O_{i-1}O_{i,i} \tag{3.2}$$

vector $O_{i-1}O_{i,i}$ has simple expression because it is represented with respect to its own frame of reference R_i. Both the vector $O_0O_{i,0}$ and $\sum_{i=1}^{n} M_0^i$ can be expressed in a recursive form, we obtain:

$$M_0^i = M_0^{i-1}.M_{i-1}^i \tag{3.3}$$

$$O_0O_{i,0} = O_0O_{i-1,0} + M_0^i.O_{i-1}O_{i,i} \tag{3.4}$$

where M_0^{i-1} is the transfer matrix from R_0 to R_{i-1}, and M_{i-1}^i is the transfer matrix

from R_i to R_{i-1}. The orientation matrix

$$M_0^n = [X_{n,o}, Y_{n,o}, Z_{n,o}] = \begin{pmatrix} X_x & Y_x & Z_x \\ X_y & Y_y & Z_y \\ X_z & Y_z & Z_z \end{pmatrix} \tag{3.5}$$

is used to determine the orientation of the frame R_n with respect to the frame R_0, and the position vector $O_0O_{n,0}$ is used to determine the position of the frame R_n with respect to the frame R_0

$$O_0O_{n,0} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \tag{3.6}$$

The fundamental direct kinematics problem for robotic manipulators is to formulate kinematic equations or what we call geometric model that transform the joint space to Cartesian space. The robot end effector is represented by position vector and orientation matrix in the three dimensional space. Therefore, a point in the joint space can be converted to a position and an orientation in the Cartesian space using the geometric model. Transformation functions relate the position and orientation of the end effector coordinate system to the base coordinate system. These transformation are very important, since a robot is servoed in the joint space, whereas most

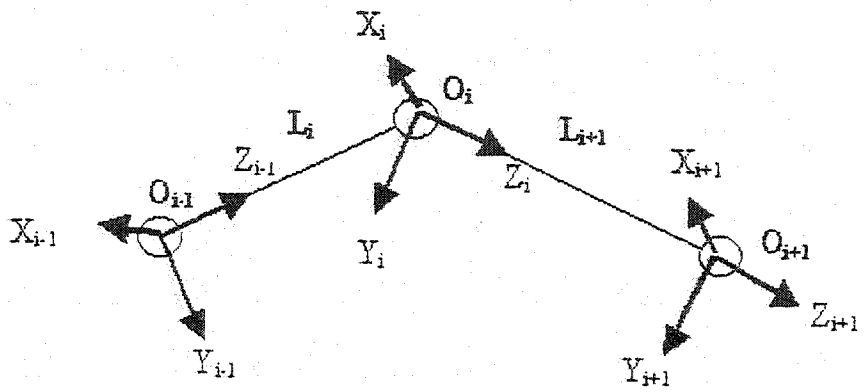

Figure 3.7: Two links with a revolute joint

of the tasks are done in the Cartesian space. The geometric model is denoted by $E = G(\theta)$, where the end effector vector E is a function of the joint variables vector θ. Given the joint variables $\theta_1, \theta_2, ...\theta_n$ we can compute the basic representation of the end effector with respect to a reference Cartesian coordinate R_0 by using the direct geometric model:

$$(\theta_1, \theta_2, ...\theta_n)^t \rightarrow \{O_0O_{n,0}(\theta), M_0^n(\theta)\} \tag{3.7}$$

The basic geometrical representation of the arm is reduced to the following expression:

$$G(\theta) = \{O_0O_{n,0}(\theta), M_0^n(\theta)\} \tag{3.8}$$

3.3 Direct Geometric Model of PUMA-560

Now we will develop a mathematical model for representing the geometric configuration of the links and joints of the PUMA-560. This manipulator arm has six links and each link can rotate with respect to a reference coordinate system. The geometric model solutions starts by assigning a link attached coordinate frame to each link of the manipulator, it then tabulates these link parameters and establishes the transformation matrix M_i^{i-1} for each link. The state of the end effector link attached frame, with respect to the base coordinate frame, can be determined by means of the following information:

- The robot hand orientation matrix $M_0^n = [X_n^0, Y_n^0, Z_n^0]$, determines the orientation of frame R_n with respect to frame R_0.
- The position vector $O_0O_{n,0}$, references the origin of R_n with respect to R_0.

The orientation matrix M_0^i is the products of the rotation matrices:

$$M_0^n = M_0^1.M_1^2.M_2^3...M_{n-1}^n \tag{3.9}$$

The position vector can be determined as follows:

$$O_0O_{n,0} = O_0O_{n-1,0} + M_0^n.O_{n-1}O_{n,n} \tag{3.10}$$

The basic concept of the mathematical description of the frames is to provide the translation and rotation characteristics of each link. Evaluation of rotation matrices and position vectors will help later to solve the inverse kinematics problem. The reference arm position required to build the geometric model for the PUMA-560 robot arm is shown in Figure 3.8. In order to develop the geometric model for the PUMA-560 robot arm we will establish the following scheme:

- Every joint is attached to a frame of reference.
- Link L_i is between frame R_{i-1} and frame R_i.
- M_i^{i+1} is used to represent the rotation between links L_i and L_{i+1}
- The orientation matrix $M_0^n = [X_n^0, Y_n^0, Zn^0]$ determines the orientation of frame R_n with respect to frame R_0.
- The position vector is given by the following expression:

$$O_0O_{i,0} = O_0O_{i-1,0} + M_0^i O_{i-1}O_{i,i} \tag{3.11}$$

The frame representation for the robot arm is shown in Figure 3.9. Functionally our robot arm, having six degrees of freedom (D.O.F) and they can be divided into two substructures which are the Transporter and the Effector parts. We have used the following topological form as indicated in [33], to describe the geometric structure

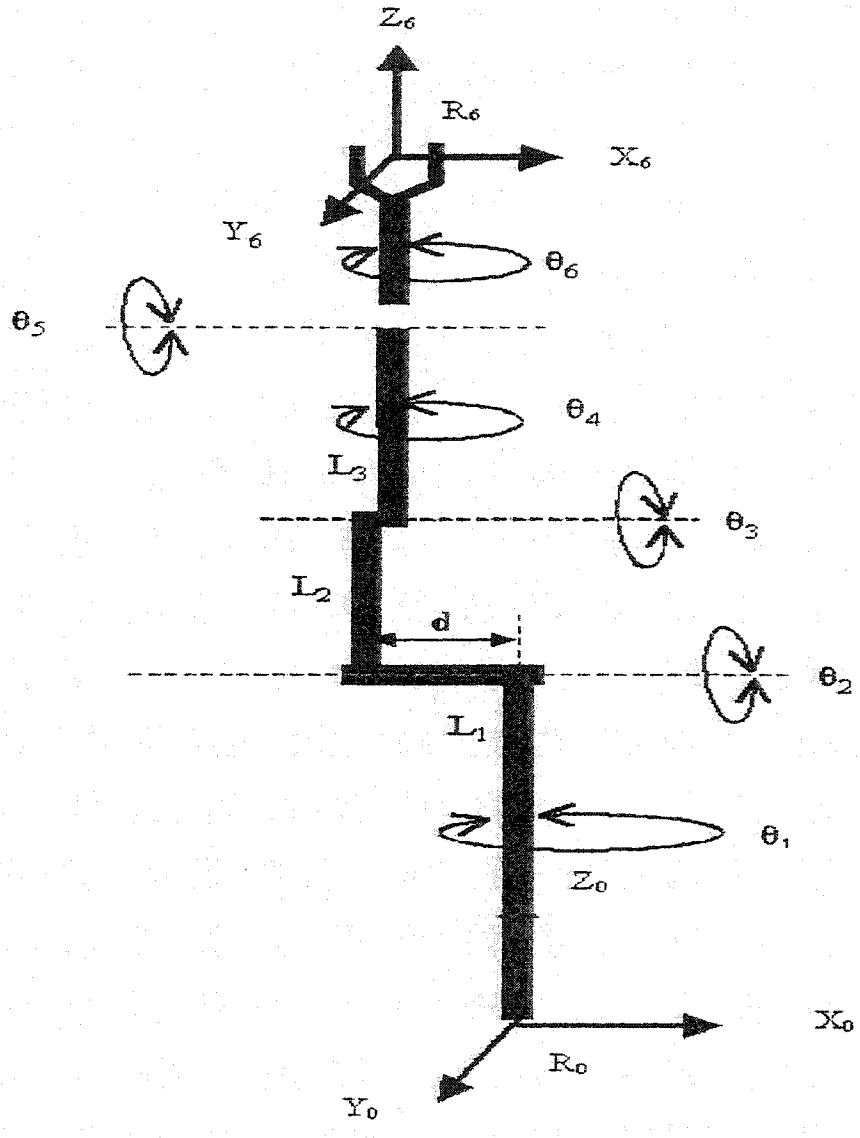

Figure 3.8: Reference arm position

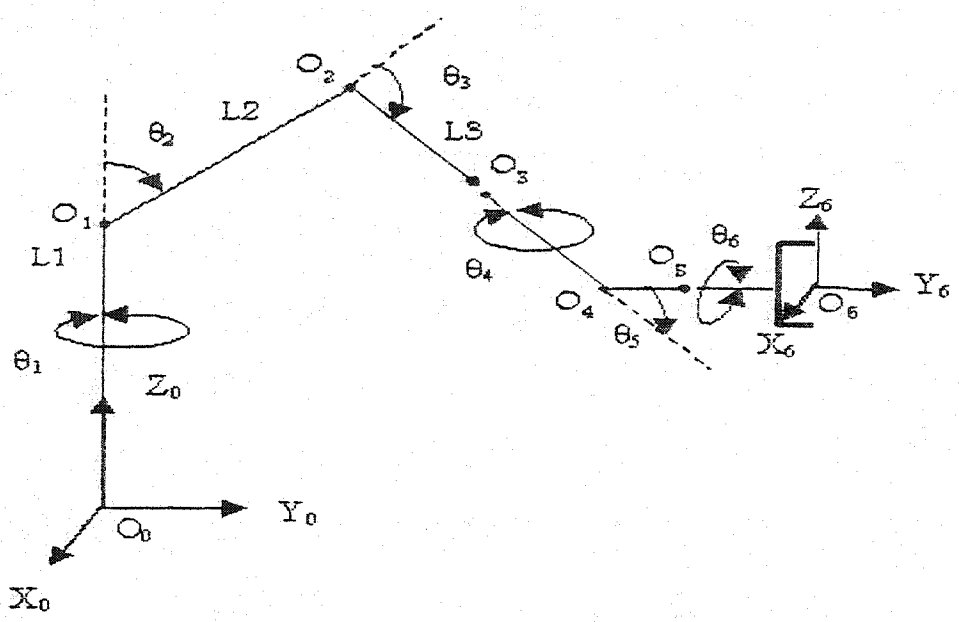

Figure 3.9: Frame representation for the PUMA-560 arm

of the arm:

The transporter part is defined by three revolute joints:

$$Linkl(ROTZ(\theta_1), Z(L1)$$

$$Link2(ROTX(\theta_2), Z(L2)$$

$$Link3(ROTX(\theta_3), Z(L3)$$

The effector part is defined by three revolute joints:

$$Link4(ROTZ(\theta_4), Z(L4)$$

$$Link5(ROTX(\theta_5), Z(L5)$$

$$Link6(ROTZ(\theta_6), Z(L6)$$

where $ROTZ(\theta_i)$ is the rotation of link i between frames R_i and R_{i-1} about the Z_{i-1} axis, and $Z(L_i)$ indicates that the link body L_i is along the Z_i vector. Consider L_1 which is revolute and defined as a rotation about the Z_o axis, and the link body

L_1 is along vector Z_1 of frame R_1.

We start by computing the transformation matrix between frames R_1 and R_0

$$M_o^1 = ROTZ(\theta_1) \tag{3.12}$$

$$M_o^1 = \begin{pmatrix} C_1 & -S_1 & 0 \\ S_1 & C_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.13}$$

The solution to the position vector $[O_0O_{1,0}]$ yields:

$$O_0O_{1,0} = M_0^1.O_0O_{1,1} \tag{3.14}$$

$$O_0O_{1,0} = \begin{pmatrix} C_1 & -S_1 & 0 \\ S_1 & C_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ \ell_1 \end{pmatrix} \tag{3.15}$$

$$O_0O_{1,0} = \begin{pmatrix} 0 \\ 0 \\ \ell_1 \end{pmatrix} \tag{3.16}$$

For the second end point O_2 we have:

$$M_0^2 = M_0^1.M_1^2 = ROTZ(\theta_1).ROTX(\theta_2) \tag{3.17}$$

$$M_0^2 = \begin{pmatrix} C_1 & -S_1 & 0 \\ S_1 & C_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & C_2 & -S_2 \\ 0 & S_2 & C_2 \end{pmatrix} \tag{3.18}$$

$$M_0^2 = \begin{pmatrix} C_1 & -S_1C_2 & S_1S_2 \\ S_1 & C_1C_2 & -C_1S_2 \\ 0 & S_2 & C_2 \end{pmatrix} \tag{3.19}$$

The position vector $O_2O_{2,0}$ is given by:

$$O_0O_{2,0} = O_0O_{1,0} + M_0^2 O_1O_{2,2} \tag{3.20}$$

The link body L_2 is along vector Z_2 of frame R_2 and is shifted by value d in the X axis direction from the link body L_1.

$$O_0O_{2,0} = O_0O_{1,0} + M_0^2 . \begin{pmatrix} 0 \\ 0 \\ \ell_1 \end{pmatrix} \tag{3.21}$$

$$O_0O_{2,0} = \begin{pmatrix} C_1.d + S_1S_2\ell_2 \\ S_1.d - C_1S_2\ell_2 \\ \ell_1 + C_2\ell_2 \end{pmatrix} \tag{3.22}$$

The link body L_3 is along vector Z_3 and rotate around X axis.

The orientation matrix M_0^3 is given by:

$$M_0^3 = M_0^2.M_2^3 = M_0^2.ROTX(\theta_3) \tag{3.23}$$

$$M_0^2 = \begin{pmatrix} C_1 & -S_1C_2 & S_1S_2 \\ S_1 & C_1C_2 & -C_1S_2 \\ 0 & S_2 & C_2 \end{pmatrix} \tag{3.24}$$

$$M_0^3 = \begin{pmatrix} C_1 & -S_1C_2 & S_1S_2 \\ S_1 & C_1C_2 & -C_1S_2 \\ 0 & S2 & C2 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & C_3 & -S_3 \\ 0 & S_3 & C_3 \end{pmatrix} \tag{3.25}$$

Given that :

$C_{23} = C_2C_3 - S_2S_3$

$S_{23} = C_2S_3 + C_3S_2$

Then

$$M_0^3 = \begin{pmatrix} C_1 & -S_1C_23 & S_1S_23 \\ S_1 & C_1C_23 & -C_1S_23 \\ 0 & S23 & C23 \end{pmatrix} \tag{3.26}$$

The position vector $O_0O_{3,0}$ is given by:

$$O_0O_{3,0} = O_0O_{2,0} + M_0^3O_2O_{3,3} \tag{3.27}$$

Then

$$O_0O_{3,0} = O_0O_{2,0} + M_0^3 . \begin{pmatrix} 0 \\ 0 \\ \ell_3 \end{pmatrix} \tag{3.28}$$

$$O_0O_{3,0} = \begin{pmatrix} C_1.d + S_1S_2\ell_2 \\ S_1.d - C_1S_2\ell_2 \\ \ell_1 + C_2\ell_2 \end{pmatrix} + \begin{pmatrix} C_1 & -S_1C_{23} & S_1S_{23} \\ S_1 & C_1C_{23} & -C_1S_{23} \\ 0 & S_{23} & C_{23} \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ \ell_3 \end{pmatrix} \tag{3.29}$$

$$O_0O_{3,0} = \begin{pmatrix} C_1.d + S_1S_2\ell_2 + S_1S_{23}\ell_3 \\ S_1.d + C_1S_2\ell_2 + C_1S_{23}\ell_3 \\ \ell_1 + C_2\ell_2 + C_{23}\ell_3 \end{pmatrix} \tag{3.30}$$

The link body L_4 is along vector Z_4 and rotate around Z axis.

For the orientation matrix we have :

$$M_0^4 = M_0^3.M_3^4 = M_0^3.ROTZ(\theta_4) \tag{3.31}$$

$$M_0^4 = \begin{pmatrix} C_1 & -S_1C_{23} & S_1S_{23} \\ S_1 & C_1C_{23} & -C_1S_{23} \\ 0 & S_{23} & C_{23} \end{pmatrix} \cdot \begin{pmatrix} C_4 & -S_4 & 0 \\ S_4 & C_4 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.32}$$

$$M_0^4 = \begin{pmatrix} C_1C_4 - S_1S_4C_{23} & -C_1S_4 - S_1C_4C_{23} & S_1S_{23} \\ S_1C_4 + C_1C_4C_{23} & -S_1S_4 + C_1C_4C_{23} & -C_1S_{23} \\ S_{23}C_4 & C_4S_{23} & C_{23} \end{pmatrix} \tag{3.33}$$

For the position vector $O_0O_{4,0}$ we have:

$$O_0O_{4,0} = O_0O_{3,0} + M_0^4O_3O_{4,4} \tag{3.34}$$

$$O_0O_{4,0} = O_0O_{3,0} + M_0^4 \cdot \begin{pmatrix} 0 \\ 0 \\ \ell_4 \end{pmatrix} \tag{3.35}$$

given that :

$$L_{34} = L_3 + L_4 \tag{3.36}$$

$$O_0O_{4,0} = \begin{pmatrix} C_1.d + S_1(S_2\ell_2 + S_23\ell_34) \\ S_1.d - C_1(S_2\ell_2 - S_{23}\ell_{34}) \\ \ell_1 + C_2\ell_2 + C_{23}\ell_{34} \end{pmatrix} \tag{3.37}$$

To simplify the following work: Let

$$M_0^4 = \begin{pmatrix} X_{x4} & Y_{x4} & Z_{x4} \\ X_{y4} & Y_{y4} & Z_{y4} \\ X_{z4} & Y_{z4} & Z_{z4} \end{pmatrix} \tag{3.38}$$

And let

$$O_0O_{4,0} = \begin{pmatrix} X_4 \\ Y_4 \\ Z_4 \end{pmatrix} \tag{3.39}$$

The link body L_5 is along vector Z_5 and rotate around X axis.

Rotation matrix M_0^5 is given by:

$$M_0^5 = M_0^4.M_4^5 = M_0^4.ROTX(\theta_5) \tag{3.40}$$

Then

$$M_0^5 = \begin{pmatrix} X_{x4} & C_5(Y_{x4}) + S_5(Z_{x4}) & -S_5(Y_{x4}) + C_5(Z_{x4}) \\ X_{y4} & C_5(Y_{y4}) + S_5(Z_{y4}) & -S_5(Y_{y4}) + C_5(Z_{y4}) \\ X_{z4} & C_5(Y_{z4}) + S_5(Z_{z4}) & -S_5(Y_{z4}) + C_5(Z_{z4}) \end{pmatrix} \tag{3.41}$$

The position vector $O_0O_{5,0}$ is expressed as in the following:

$$O_0O_{5,0} = O_0O_{4,0} + M_0^5O_4O_{5,5} \tag{3.42}$$

$$O_0O_{5,0} = O_0O_{4,0} + M_0^5.\begin{pmatrix} 0 \\ 0 \\ \ell_5 \end{pmatrix} \tag{3.43}$$

$$O_0O_{5,0} = \begin{pmatrix} X_4 - S_5(-C_1S_4 - S_1C_4C_23) + C_5(S_1S_23).\ell_5 \\ Y_4 - S_5(-S_1S_4 + C_1C_4C_23) + C_5(-C_1S_23).\ell_5 \\ Z_4 - S_5(C_4S_23) + C_5(C_23).\ell_5 \end{pmatrix} \tag{3.44}$$

Finally, the link body L_6 is along vector Z_6 and rotate around Z axis. The basic orientation matrix M_0^6 is computed as follows:

$$M_0^6 = M_0^5.M_5^6 = M_0^5.ROTZ(\theta_6) \tag{3.45}$$

$$M_0^6 = \begin{pmatrix} X_x & Y_x & Z_x \\ X_y & Y_y & Z_y \\ X_z & Y_z & Z_z \end{pmatrix} \tag{3.46}$$

The final components of the orientation matrix are:

$$X_x = C_6(C_1C_4 - S_1S_4C_{23}) + S_6(-C_1C_5S_4 - S_1C_4C_5C_{23}) + S_1S_5S_23)$$

$$X_y = C_6(S_1C_4 + C_1C_4C_{23}) + S_6(-S_1S_4C_5 + C_1C_4C_5C_23) - C_1S_5S_23)$$

$$X_z = C_6(S_{23}C_4) + S_6(C_4C_5S_{23} + S_5C_{23})$$

$$Y_x = -S_6(C_1C_4 - S_1S_4C_{23}) + C_6(-C_1C_5S_4 - S_1C_4C_5C_{23}) + S_1S_5S_23)$$

$$Y_y = -S_6(S_1C_4 + C_1C_4C_{23}) + C_6(-S_1S_4C_5 + C_1C_4C_5C_23) - C_1S_5S_23)$$

$$Y_z = -S_6(S_{23}C_4) + C_6(C_4C_5S_{23} + S_5C_{23})$$

$$Z_x = -S_5(-C_1S_4 - S_1C_4C_5C_{23}) + S_1C_5S_23)$$

$$Z_y = -S_5(-S_1S_4 + C_1C_4C_5C_23) - C_1C_5S_23)$$

$$Z_z = -S_5(C_4S_{23} + C_5C_{23})$$

The end effector position vector is given by:

$$O_0O_{6,0} = O_0O_{5,0} + M_0^6O_5O_{6,6} \tag{3.47}$$

$$O_0O_{6,0} = O_0O_{5,0} + M_0^6.\begin{pmatrix} 0 \\ 0 \\ \ell_6 \end{pmatrix} \tag{3.48}$$

$$O_0O_{6,0} = \begin{pmatrix} X_4 + (L_5 + L_6).Z_x \\ Y_4 + (L_5 + L_6).Z_y \\ Z_4 + (L_5 + L_6).Z_z \end{pmatrix} \tag{3.49}$$

The final position of the end effector is given by :

$$X = X_4 + (L_5 + L_6).Z_x$$

$$Y = Y_4 + (L_5 + L_6).Z_y$$

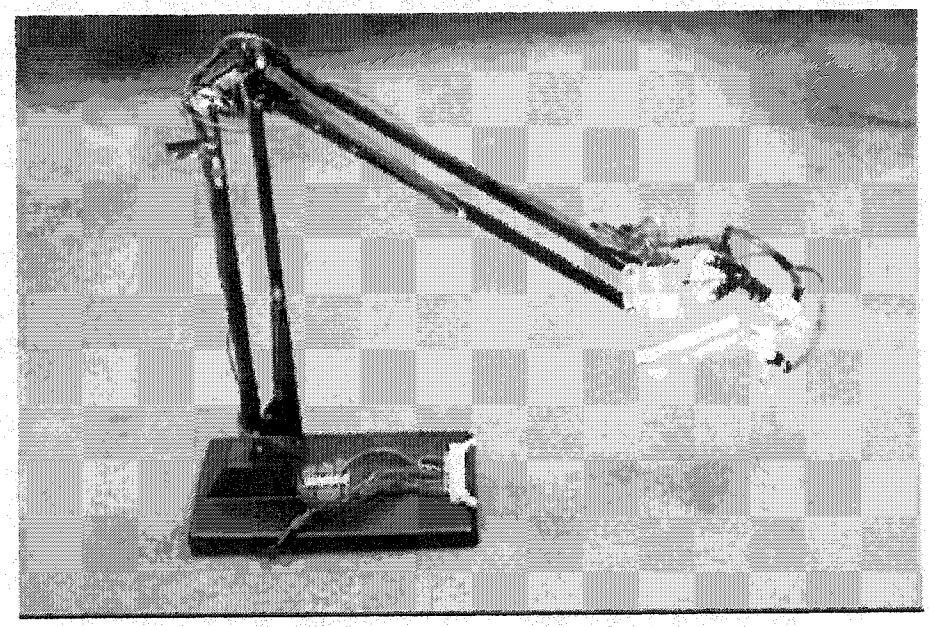

Figure 3.10: The master arm

$$Z = Z_4 + (L_5 + L_6).Z_z$$

3.4 The Geometric Model for Master Arm

Our master arm is shown in Figure 3.10. It has six degrees of freedom with all revolute joints. All the joints are equipped with potentiometers (sensors) which are used to measure the angles between joints. In addition to these sensors, the holder at the end effector part of the master arm is attached with a Stop/Start switch and mode selector switches. The stop/start button is used to enable/disable the master arm, and hence can be used by the operator to disable the system temporarily, move the master arm to a comfortable location, and then re-enable the system. The mode selector switch is used to select the mode of operation. The geometric model allows

mapping trajectories of the hand effector that are described in the joint space, into the corresponding trajectories in the Cartesian space. The position and the orientation of the effector can be totally determined by means of the following information:

1. The hand center or vector $O_nO_{n,0}$ which references the origin of R_n with respect to R_0:

$$O_nO_{n,0} = \sum_{i=1}^{n} O_{i-1,0}O_{i,0} \tag{3.50}$$

2. The hand orientation matrix $M_0^n = [X_{n,0}, Y_{n,0}, Z_{n,0}]$ determines the orientation of frame R_n with respect to frame R_0.

The frame representation of the master arm is shown in Figure 3.11 and the following topological form is used to describe the geometric structure of the master arm :

$$Link1(ROTZ(\theta_1), Z(L1)); L_1 = 100mm$$
$$Link2(ROTX(\theta_2), Z(L2)); L_2 = 360mm$$
$$Link3(ROTX(\theta_3), Y(L3)); L_3 = 350mm$$
$$Link4(ROTY(\theta_4), Y(L4)); L_4 = 20mm$$
$$Link5(ROTX(\theta_5), Y(L5)); L_5 = 50mm$$
$$Link6(ROTY(\theta_6), Y(L6)); L_6 = 20mm$$

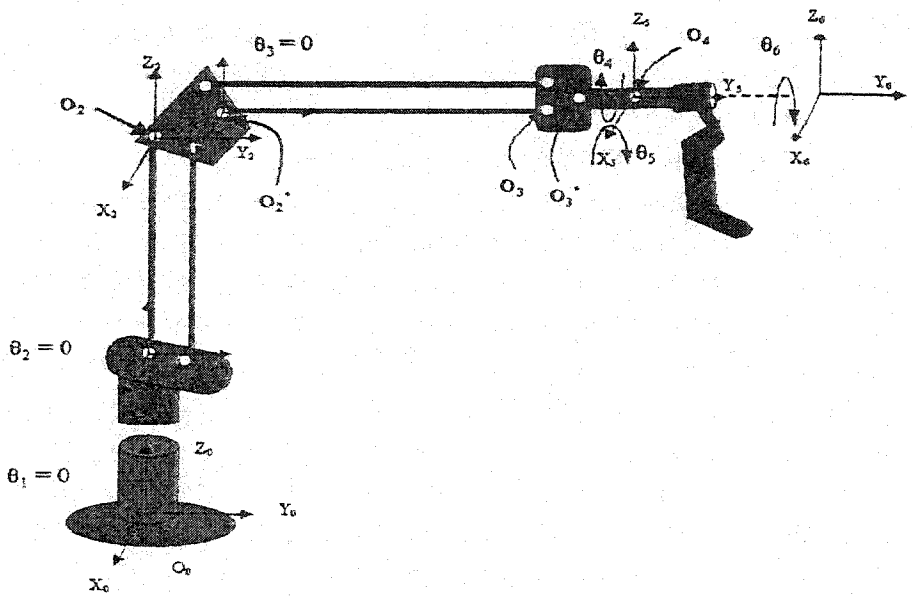

Figure 3.11: Frame representation for the master arm

The link body L_1 is along the Z_1 vector and rotate around the Z axis.

For the first end point O_1 in the master arm we have:

$$O_0O_{1,0} = M_0^1.O_0O_{1,1} \tag{3.51}$$

M_0^1 is the rotation matrix between R_0 and R_1 frames, and is given by:

$$M_0^1 = ROTZ(\theta_1) \tag{3.52}$$

$$M_o^1 = \begin{pmatrix} C_1 & -S_1 & 0 \\ S_1 & C_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.53}$$

The position vector is given by:

$$O_0O_{1,0} = M_0^1 . O_0O_{1,1} \tag{3.54}$$

where $O_0O_{1,1}$ is simply expressed as:

$$O_0O_{1,1} = \begin{pmatrix} 0 \\ 0 \\ \ell_1 \end{pmatrix} \tag{3.55}$$

Therefore, position vector $O_0O_{1,0}$ is given by:

$$O_0O_{1,0} = \begin{pmatrix} C_1 & -S_1 & 0 \\ S_1 & C_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ \ell_1 \end{pmatrix} \tag{3.56}$$

$$O_0O_{1,0} = \begin{pmatrix} 0 \\ 0 \\ \ell_1 \end{pmatrix} \tag{3.57}$$

The link body L_2 is along vector Z_2 of frame R_2 and rotate around X_1

$$M_0^2 = M_0^1 . M_1^2 = ROTZ(\theta_1) . ROTX(\theta_2) \tag{3.58}$$

$$M_0^2 = \begin{pmatrix} C_1 & -S_1 & 0 \\ S_1 & C_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & C_2 & -S_2 \\ 0 & S_2 & C_2 \end{pmatrix} \tag{3.59}$$

$$M_0^2 = \begin{pmatrix} C_1 & -S_1C_2 & S_1S_2 \\ S_1 & C_1C_2 & -C_1S_2 \\ 0 & S_2 & C_2 \end{pmatrix} \tag{3.60}$$

The position vector $O_2O_{2,0}$ is given by:

$$O_0O_{2,0} = O_0O_{1,0} + M_0^2 . \begin{pmatrix} 0 \\ 0 \\ \ell_2 \end{pmatrix} = \begin{pmatrix} S_1S_2\ell_2 \\ -C_1S_2\ell_2 \\ \ell_1 + C_2\ell_2 \end{pmatrix} \tag{3.61}$$

We found that the link L_3 drifts in the Y and Z directions from the origin O_2 as shown in Figure 3.12. These drifts are measured as:

$$y_a = 45mm$$

$$z_a = 12.5mm$$

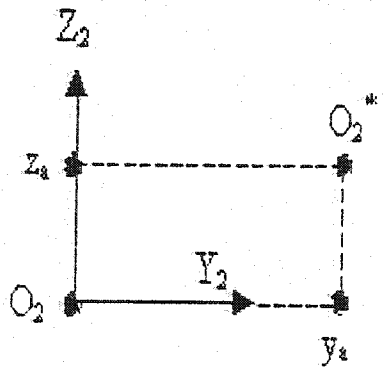

Figure 3.12: O_2^* relative to O_2

The orientation of the mechanical piece holding O_2 and O_2^* is fixed. the modified position vector $O_2O_{2,0}^*$ is given by:

$$O_0O_{2,0}^* = O_0O_{2,0} + M_0^2 O_2O_{2,2}^* \tag{3.62}$$

where

$$O_2O_{2,2}^* = \begin{pmatrix} 0 \\ y_a \\ z_a \end{pmatrix} \tag{3.63}$$

$$O_0O_{2,0}^* = \begin{pmatrix} X_2^* \\ Y_2^* \\ Z_2^* \end{pmatrix} = \begin{pmatrix} S_1S_2(\ell_2 + z_a) - S_1C_2y_0 \\ -C_1S_2(\ell_2 + z_a) + C_1C_2y_0 \\ \ell_1 + C_2(\ell_2 + z_a) + S_2y_0 \end{pmatrix} \tag{3.64}$$

The link body L_3 is along vector Y_3 and rotate around X axis.

Frame R_3 depends only on θ_l and θ_3. Therefore, The orientation matrix M_0^3 is given by:

$$M_0^3 = M_0^1.M_2^3 = ROTZ(\theta_1).ROTX(\theta_3) \tag{3.65}$$

$$M_0^3 = \begin{pmatrix} C_1 & -S_1C_3 & S_1S_3 \\ S_1 & C_1C_3 & -C_1S_3 \\ 0 & S_3 & C_3 \end{pmatrix} \tag{3.66}$$

The position vector $O_0O_{3,0}$ is given by:

$$O_0O_{3,0} = O_0O_{2,0}^* + M_0^3O_2O_{3,3} \tag{3.67}$$

$$O_0O_{3,0} = O_0O_{2,0}^* + M_0^3.\begin{pmatrix} 0 \\ \ell_3 \\ 0 \end{pmatrix} \tag{3.68}$$

The link L_4 drifts in the Y and Z directions from the origin O_3 as shown in Figure 3.13, which are measured as:

$$y_b = 40mm$$

$$z_b = 22.5mm$$

The modified position vector $O_0O_{3,0}^*$ is expressed as follow:

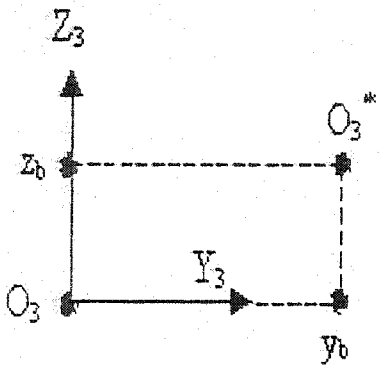

Figure 3.13: O_3^* relative to O_3

$$O_0O^*_{3,0} = O_0O^*_{2,0} + M^3_0 . \begin{pmatrix} 0 \\ \ell_3 + y_b \\ z_b \end{pmatrix} \tag{3.69}$$

$$O_0O^*_{3,0} = \begin{pmatrix} X^*_3 \\ Y^*_3 \\ Z^*_3 \end{pmatrix} = \begin{pmatrix} X^*_2 - S_1C_3(\ell_3 + y_b) + S_1S_3z_1 \\ Y^*_2 + C_1C_3(\ell_3 + y_b) - C_1S_3z_1 \\ Z^*_2 + S_3(\ell_3 + y_b) + C_3z_1 \end{pmatrix} \tag{3.70}$$

The link body L_4 is along vector Y_4 and rotate around Y axis.

The orientation matrix M^4_0 depends only on θ_1 and θ_4:

$$M^4_0 = M^1_0 . M^4_3 = ROTZ(\theta_1) . R0TY(\theta_4) \tag{3.71}$$

$$M_0^4 = \begin{pmatrix} C_1C_4 & -S_1 & C_1S_4 \\ S_1C_4 & C_1 & S_1S_4 \\ -S_4 & 0 & C_4 \end{pmatrix} \tag{3.72}$$

for the position vector $O_0O_{4,0}$ we have:

$$O_0O_{4,0} = O_0O^*_{3,0} + M_0^4 O_3O^*_{4,4} \tag{3.73}$$

$$O_0O_{4,0} = O_0O^*_{3,0} + M_0^4 . \begin{pmatrix} 0 \\ \ell_4 \\ 0 \end{pmatrix} \tag{3.74}$$

$$O_0O_{4,0} = \begin{pmatrix} X_4 \\ Y_4 \\ Z_4 \end{pmatrix} = \begin{pmatrix} X^*_3 - S_1\ell_4 \\ Y^*_3 + C_1\ell_4 \\ Z^*_3 \end{pmatrix} \tag{3.75}$$

The link body L_5 is along vector Y_5 and rotate around X axis.

The orientation matrix M_0^5 is given by:

$$M_0^5 = M_0^4 . M_4^5 = M_0^4 . ROTX(\theta_5) \tag{3.76}$$

$$M_0^5 = \begin{pmatrix} C_1C_4 & -S_1 & C_1S_4 \\ S_1C_4 & C_1 & S_1S_4 \\ -S_4 & 0 & C_4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & C5 & -S5 \\ 0 & S5 & C5 \end{pmatrix} \tag{3.77}$$

$$M_0^5 = \begin{pmatrix} C_1C_4 & -S_1C_5 + C_1S_4S_5 & S_1S_5 + C_1S_4C_5 \\ S_1C_4 & C_1C_5 + S_1S_4S_5 & -C_1S_5 + S_1S_4C_5 \\ -S_4 & C_4S_5 & C_4C_5 \end{pmatrix} \tag{3.78}$$

The position vector $O_0O_{5,0}$ is computed as following:

$$O_0O_{5,0} = O_0O_{4,0} + M_0^5 O_4O_{5,5} \tag{3.79}$$

$$O_0O_{5,0} = O_0O_{4,0} + M_0^5 \begin{pmatrix} 0 \\ \ell_5 \\ 0 \end{pmatrix} \tag{3.80}$$

$$O_0O_{5,0} = O_0O_{4,0} + M_0^5 \begin{pmatrix} 0 \\ \ell_5 \\ 0 \end{pmatrix} \tag{3.81}$$

$$O_0O_{5,0} = \begin{pmatrix} X_4 + (-S_1C_5 + C_1S_4S_5).\ell_5 \\ Y_4 + (C_1C_5 + S_1S_4S_5)\ell_5 \\ Z_4 + C_4S_5\ell_5 \end{pmatrix} \tag{3.82}$$

The link body L_6 is along vector Y_6 and rotate around Y axis.

The final orientation matrix M_0^6 is given by:

$$M_0^6 = M_0^5.M_5^6 = M_0^5.ROTY(\theta_6) \tag{3.83}$$

$$M_0^6 = M_0^5 \cdot \begin{pmatrix} C_6 & 0 & S_6 \\ 0 & 1 & 0 \\ -S_6 & 1 & C_6 \end{pmatrix} \tag{3.84}$$

The final expression for the orientation matrix is given by:

$$M_0^6 = \begin{pmatrix} X_x & Y_x & Z_x \\ X_y & Y_y & Z_y \\ X_z & Y_z & Z_z \end{pmatrix} \tag{3.85}$$

The components of the orientation matrix are given below:

$$X_x = C_1C_4C_6 - S_6(S_1S_5 + C_1S_4C_5)$$

$$X_y = S_1C_4C_6 - S_6(-C_1S_5 + C_1S_4C_5)$$

$$X_Z = -S_4C_6 - S_6(C_4C_5)$$

$$Y_x = -S_1C_5 + C_1S_4S_5$$

$$Y_y = C_1C_5 + S_1S_4S_5$$

$$Y_z = C_4S_5$$

$$Z_x = C_1C_4S_6 + C_6(S_1S_5 + C_1S_4C_5)$$

$$Z_y = S_1C_4S_6 + C_6(-C_1S_5 + C_1S_4C_5)$$

$$Z_z = -S_4S_6 + C_6(C_4C_5)$$

The end effector position vector is given by:

$$O_0O_{6,0} = O_0O_{5,0} + M_0^6O_5O_{6,6} \tag{3.86}$$

$$O_0O_{6,0} = O_0O_{5,0} + M_0^6.\begin{pmatrix} 0 \\ \ell_6 \\ 0 \end{pmatrix} \tag{3.87}$$

$$O_0O_{6,0} = \begin{pmatrix} X_4 + Y_x.\ell_6 \\ Y_4 + Y_y.\ell_6 \\ Z_4 + Y_z.\ell_6 \end{pmatrix} \tag{3.88}$$

The $O_0O_{6,0}$ and M_0^6 components of the position vector of the end effector for the master arm are given below:

$$X = X_4 + Y_x.\ell_6$$

$$Y = Y_4 + Y_y.\ell_6$$

$$Z = Z_4 + Y_z.\ell_6$$

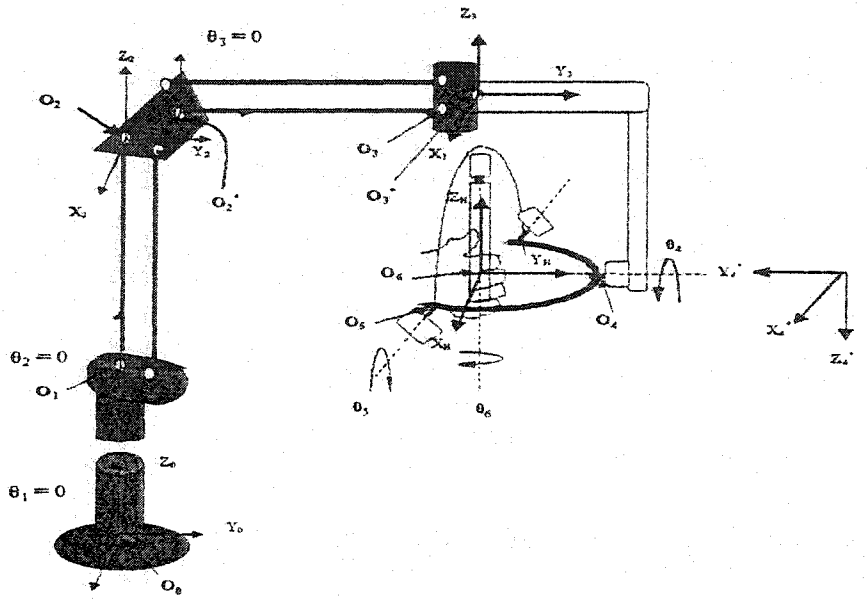

Figure 3.14: The new structure of the master arm

The Geometric Model for the New Structure

We have developed the structure of the master arm to improve the operator performance and to overcome the problems of the previous structure. You may refer to Section 5.2 to see more details about this subject. In this Section we will discuss the changes in the geometric model for the new structure of the master arm. The transporter part of the master arm is the same as in Section 3.4. The new structure of the master arm is shown in Figure 5.10. To obtain the geometric model for the new structure, we have to express vectors $O_0O_{6,0}$, X_6, Y_6 and Z_6 with respect to the absolute coordinates system R_0. Since the transporter part remains the same, we can use the position vector $O_0O^*_{3,0}$ and the orientation matrix M^3_0 which have been

computed and discussed in this chapter. The orientation matrix M_0^3 is given by:

$$M_0^3 = \begin{pmatrix} C_1 & -S_1C_3 & S_1S_3 \\ S_1 & C_1C_3 & -C_1S_3 \\ 0 & S_3 & C_3 \end{pmatrix} \tag{3.89}$$

and the position vector $O_0O_{3,0}^*$ is given by:

$$O_0O_{3,0}^* = O_0O_{2,0}^* + M_0^3 . \begin{pmatrix} 0 \\ \ell_3 + y_b \\ z_b \end{pmatrix} \tag{3.90}$$

$$O_0O_{3,0}^* = \begin{pmatrix} X_3^* \\ Y_3^* \\ Z_3^* \end{pmatrix} = \begin{pmatrix} X_2^* - S_1C_3(\ell_3 + y_b) + S_1S_3z_1 \\ Y_2^* + C_1C_3(\ell_3 + y_b) - C_1S_3z_1 \\ Z_2^* + S_3(\ell_3 + y_b) + C_3z_1 \end{pmatrix} \tag{3.91}$$

The operator hand is at the origin of the concurrent rotations and below the frame R_3^* by small drifts in -Z and y directions as shown in Figure 3.15, which are measured as $z_c = -150mm$ and $y_c = 60mm$. So we compute the position vector of the end effector as following:

$$O_0O_{6,0} = O_0O_{3,0}^* + M_0^1O_3O_{6,3} \tag{3.92}$$

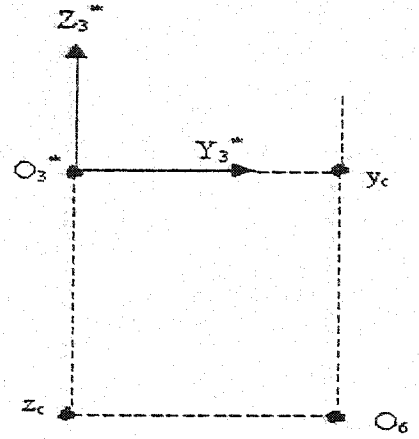

Figure 3.15: O_6 relative to O_3^*

where

$$O_3O_{6,3} = \begin{pmatrix} 0 \\ y_c \\ z_c \end{pmatrix} \tag{3.93}$$

and

$$O_0O_{3,0}^* = \begin{pmatrix} X_3^* \\ Y_3^* \\ Z_3^* \end{pmatrix} \tag{3.94}$$

The X, Y, and Z coordinates of the position vector of the end effector are:

$$X = X_3^* - S_1.y_c$$

$$Y = Y_3^* + C_1.y_c$$

$$Z = Z_3^* + z_c$$

Now we will proceed to find the orientation matrix of the system. Because of the new structure of the master arm the frame R_4^* rotated relative to frame R_3 about X by 180 degrees and about Z by 180 degrees as shown in Figure 5.10 where $\{X_4^*, Y_4^*, Z_4^*\}$ $= \{X_3, -Y_3, -Z_3\}$. We use the following expression for the $M_0^{4^*}$ orientation matrix:

$$M_0^{4^*} = M_0^3 . \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \tag{3.95}$$

$$M_0^{4^*} = \begin{pmatrix} C_1 & S_1 & 0 \\ S_1 & -C_1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \tag{3.96}$$

The orientation matrix M_0^4 is given by:

$$M_0^4 = M_0^{4^*} . ROTY(\theta_4) \tag{3.97}$$

$$M_0^4 = \begin{pmatrix} C_1C_4 & S_1 & C_1S_4 \\ S_1C_4 & -C_1 & S_1S_4 \\ S_4 & 0 & -C_4 \end{pmatrix} \tag{3.98}$$

The orientation matrix M_0^5 is given by:

$$M_0^5 = M_0^4.M_4^5 = M_0^4.ROTX(\theta_5) \tag{3.99}$$

$$M_0^5 = \begin{pmatrix} C_1C_4 & S_1 & C_1S_4 \\ S_1C_4 & -C_1 & S_1S_4 \\ S_4 & 0 & -C_4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & C5 & -S5 \\ 0 & S5 & C5 \end{pmatrix} \tag{3.100}$$

$$M_0^5 = \begin{pmatrix} C_1C_4 & S_1C_5 + C_1S_4S_5 & -S_1S_5 + C_1S_4C_5 \\ S_1C_4 & -C_1C_5 + S_1S_4S_5 & C_1S_5 + S_1S_4C_5 \\ -S_4 & -C_4S_5 & -C_4C_5 \end{pmatrix} \tag{3.101}$$

The final orientation matrix M_0^6 is given by:

$$M_0^6 = M_0^5.M_5^6 = M_0^5.ROTZ(\theta_6) \tag{3.102}$$

$$M_0^6 = M_0^5. \begin{pmatrix} C_6 & -S_6 & 0 \\ S_6 & C_6 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.103}$$

The final expression for the orientation matrix is given by:

$$M_0^6 = \begin{pmatrix} X_x & Y_x & Z_x \\ X_y & Y_y & Z_y \\ X_z & Y_z & Z_z \end{pmatrix} \tag{3.104}$$

the components of this matrix are given below:

$$X_x = C_1C_4C_6 + S_6(S_1C_5 + C_1S_4S_5)$$

$$X_y = S_1C_4C_6 + S_6(-C_1C_5 + S_1S_4S_5)$$

$$X_Z = S_4C_6 - S_6(C_4S_5)$$

$$Y_x = -C_1C_4S_6 + C_6(S_1C_5 + C_1S_4S_5)$$

$$Y_y = -S_1C_4S_6 - C_6(C_1C_5 - C_1S_4S_5)$$

$$Y_z = -S_4S_6 - C_6(C_4S_5)$$

$$Z_x = -S_1S_5 + C_1S_4C_5$$

$$Z_y = C_1C_5 + S_1S_4C_5$$

$$Z_z = -C_4C_5$$

In this chapter we use the position vector $O_0O_{6,0}$ and the orientation matrix M_0^6 to geometrically represent the robot and the master arms in the three dimensional space. The direct geometric model is used to convert a point in the joint space to a position and orientation in the Cartesian space. We have developed a computer program that directly reads the joint angles of both master arm and robot arm and generates the direct geometric solution. The master arm has six potentiometers for joint angles measurements, while the joint angles of the robot arm are determined using incremental optical encoders and potentiometers. The potentiometers that are used in the master arm are normal type available in the market, in which their accuracy is acceptable, but they can be replaced by a better type to improve the accuracy of angle readings. The replacement of the old master arm with the new one does not add any major changes in the geometric model. On the other hand, it improves the operation and performance of the human operator.

Chapter 4

VISION BASED MASTER ARM

This chapter deals with a vision based system which enables the operator to move the robot arm by moving a 3D object without using any connection to the computer. The 3D object is composed of four balls of different colors, as shown in Figure 4.1, connected by rods of the same length and orthogonal to each other. By using the 3D object, we can obtain six parameters, 3 for the position and 3 for the orientation. Since there is no electrical connection or cables between the computer and the 3D object, the operator can work in a more comfortable environment. Fukumoto et al.1992 have proposed a stereo vision system where the user can point a place on the computer's screen by his hand and give some commands by hand gesture. However, such conventional vision systems still impose some restrictions. They require camera calibration which limits the user motion. The absolute position and orientation of the hand in the space are used for gesture recognition, which means that the operator

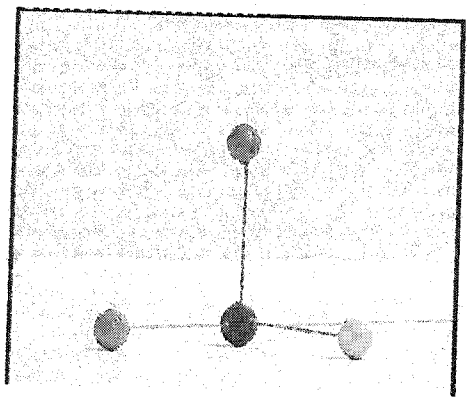

Figure 4.1: The 3D object

should not move his body from the initial position at the calibration. However, the operator cannot expect how the system will work when he moves his body from the initial position. Y. Kuno [28, 29] proposed an interface method that solved the above issue by applying un-calibrated stereo vision. Their system is based on the multiple view affine invariance theory. It calculates hand positions as invariant coordinates in the basis derived from four points on the user's body in what they call a user-centered frame so that the operator can move the object forward and backward relative to his body by moving his hand forward and backward even if he changed his body position.

We proposed a system that avoids taking the operator's position into account Our system recognizes the translation and rotation of the 3D object regardless of the position of the operator. We also use the multiple view affine invariance theory.The position and orientation of the 3D object is calculated relative to the same object in a predefined position. The position of the balls are calculated as invariant coordinates

in the coordinate system with the three basis vectors defined by the four points. The position and the orientation of the 3D object is calculated in two ways: One way is to calculate the absolute position and orientation of the object by considering the initial position as the reference frame. In the second approach the position and the orientation of the object are computed relative to the position and orientation of the object in the previous frame. This incremental motion is mapped directly to the tool frame of the robot arm. The user can move the robot arm by moving the 3D object regardless of his position.This system does not need any camera calibration because the camera parameters do not affect affine invariance.

The point detection and tracking is essential part in our system. We are using the 3D object as the master arm with six degrees of freedom. In order to get accurate and robust 3D position and orientation measurements, there are many issues that must be considered. These are appropriate detection methods, suitable tracking approaches, and a good mechanism for boundary detection. In addition to that, the center of the ball should be computed precisely. The use of color information to detect objects is found to be better and gives wider range in terms of color selection compared to the gray image based techniques. In the computer vision, the detection of a colored object is known to play essential rule in extracting specific information from the scene therefore we have selected the RGB color space for the image processing and color detection.

4.1 RGB Color Space

Due to the structure of human eye all colors are seen as variable combination of the three primary colors: Red, Green, and Blue. There are different format for colors in video cameras, such as YUV, HIS,..etc. We have used the 24 bits RGB format. Ideally speaking any primary color like red, green or blue should consist of one component only. For example Red color should have RGB coordinates = (255,0,0). Practically speaking we analyzed different images taken by different cameras and we found that the colors are not stable under different conditions. For example, we have examined the red ball and we found that the RGB value for that ball is as shown in Figure 4.2. We can see from the Figure that the G and B values for the red color are not zeros as we expected. There are various reasons for the problem of variation in the values of RGB components, like light reflection, color saturation, camera configuration, external noise,..etc. Besides the RGB color space, there are some other 3-D color spaces, like YUV, YCBCR, HIS,..etc., but in general using any algorithm other than RGB needs to compensate for the computational complexity of the transformation. For a high-speed image application, the extra computational overhead and delay may not be feasible. Therefore, processing such as edge detection based on the raw RGB coordinates will have an advantage. The existence of a color edge implies changes in terms of chromaticity or luminance or both. However monochrome (i.e. gray scale) edges only take into account changes in luminance.

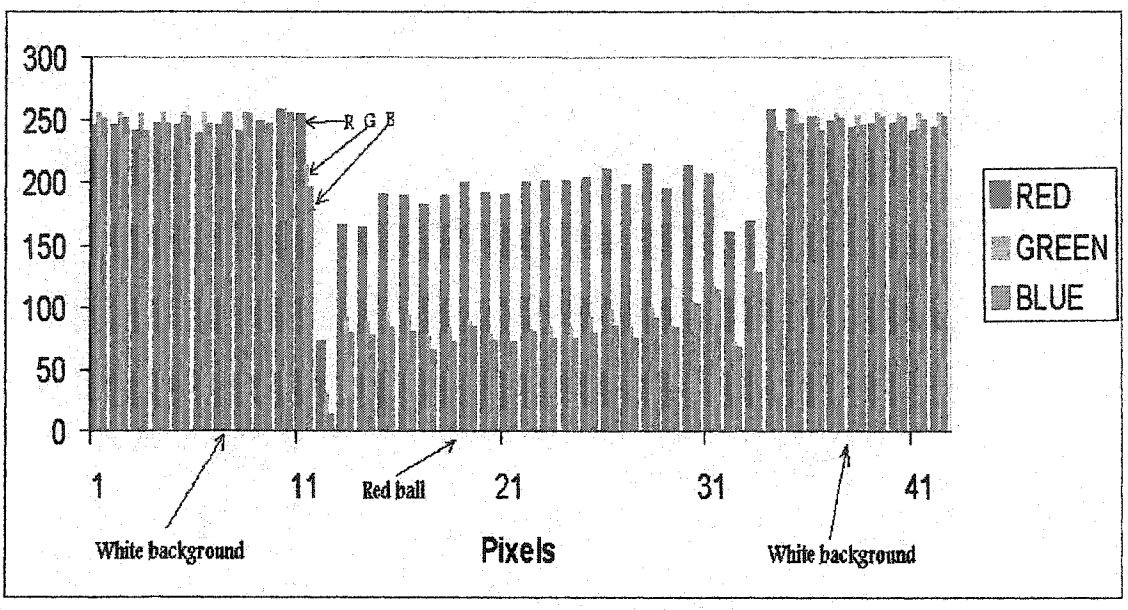

Figure 4.2: The primary colors: R,G, and B

Therefore, the edges detected in a monochrome image may not correspond to the set of edges existing in a color image, because of regions indistinguishable in terms of their luminance but differ in terms of their chromaticity or vice versa. In the past, edge features are generally obtained by applying special filters and gradient operators to gray level images with the aim of minimizing false detection and enhancing the noise rejection capability. The use of color information had been suggested and proven to perform better than techniques that operate on gray image alone.

4.2 Vision Program

The vision program is divided into two parts. The first one is about the methods that are used to detect and track the balls. It is called detection and tracking part

and is explained in Section 4.3. In Section 4.4, we explain the second part of the program which deals with the techniques that are used to measure the 3D position and orientation of the balls using the information gathered from the 2D images captured by the video cameras. In Section 4.5, we will see the details of the vision system. Before we go to the details of the program, we will introduce the user interface in the following section.

4.2.1 User Interface

A simple, but informative user interface was designed for the vision program. The vision program interface is shown in Figure 4.3. This interface hide the complexity of the programming and computation from the user and provides him the necessary information only. The interface consists of a window for image display in the upper left part. This window is an interactive, where it can be used for analyzing the features of the image. This is simply done by clicking the mouse over the area which is required to be analyzed and the results for the mouse click can be shown at the bottom left side of the program interface as shown in Figure 4.3. The data that can be displayed to the user includes the RGB values, the MVD, the MDD, and the HD function. Details of these functions are discussed in Section 4.3. The user can stop the program any time to analyze any point on the image and he can save the results to a file for later use. The image window is also used for the supervisory teaching as we will see in Section 4.3. The position of the balls on the 2D image

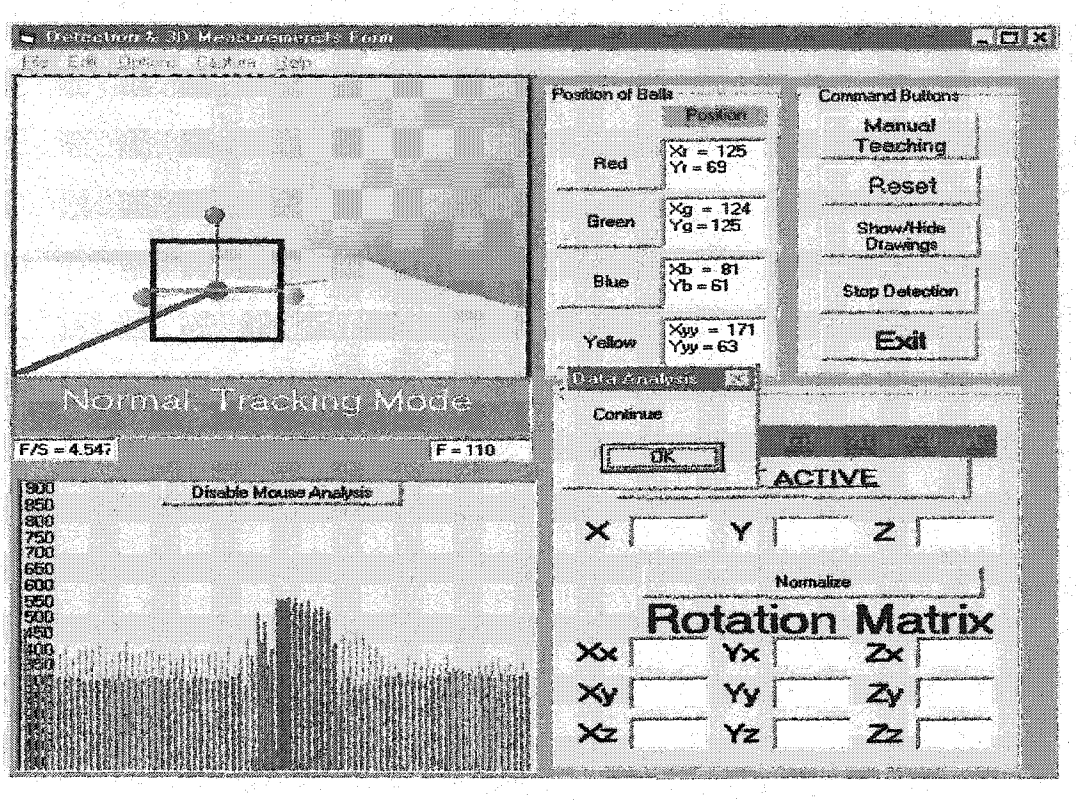

Figure 4.3: User Interface

space as well as the position and orientation in the 3D space are displayed on the right side. The 3D position and orientation of the balls can be computed only if we have the other camera connected to the system. The other camera is connected to another PC, called the client-PC. The client-PC has the same environment as this environment except that it does not have the 3D computation module. Details of the 3D computation are described in Section 4.4.

4.3 Part I: Detection and Tracking

In the first part of the vision program we will introduce the methods that are used for detecting and tracking the balls. The flow chart in Figure 4.4 describes the program structure.

Initialization

The vision program will start by calling the supervisory teaching mode. The 3D structure should be set to the initial position. The 3D structure as shown in Figure 4.1, consists of four balls: red ball which is at the center of the structure, yellow ball, green ball and blue ball. In the supervisory mode, the human operator will respond to the program by clicking the mouse over the ball that is requested by the program as shown in Figure 4.5. The program will store the location and the feature parameters of the point that was marked by the operator in global variables to be used later as a guide for that ball features and characteristics.

New Frame

With every new frame, the detection flags for the points(balls) will be checked, if all detection flags are true then the mode flag will be set to tracking mode, otherwise the mode flag will be set to the detection mode.

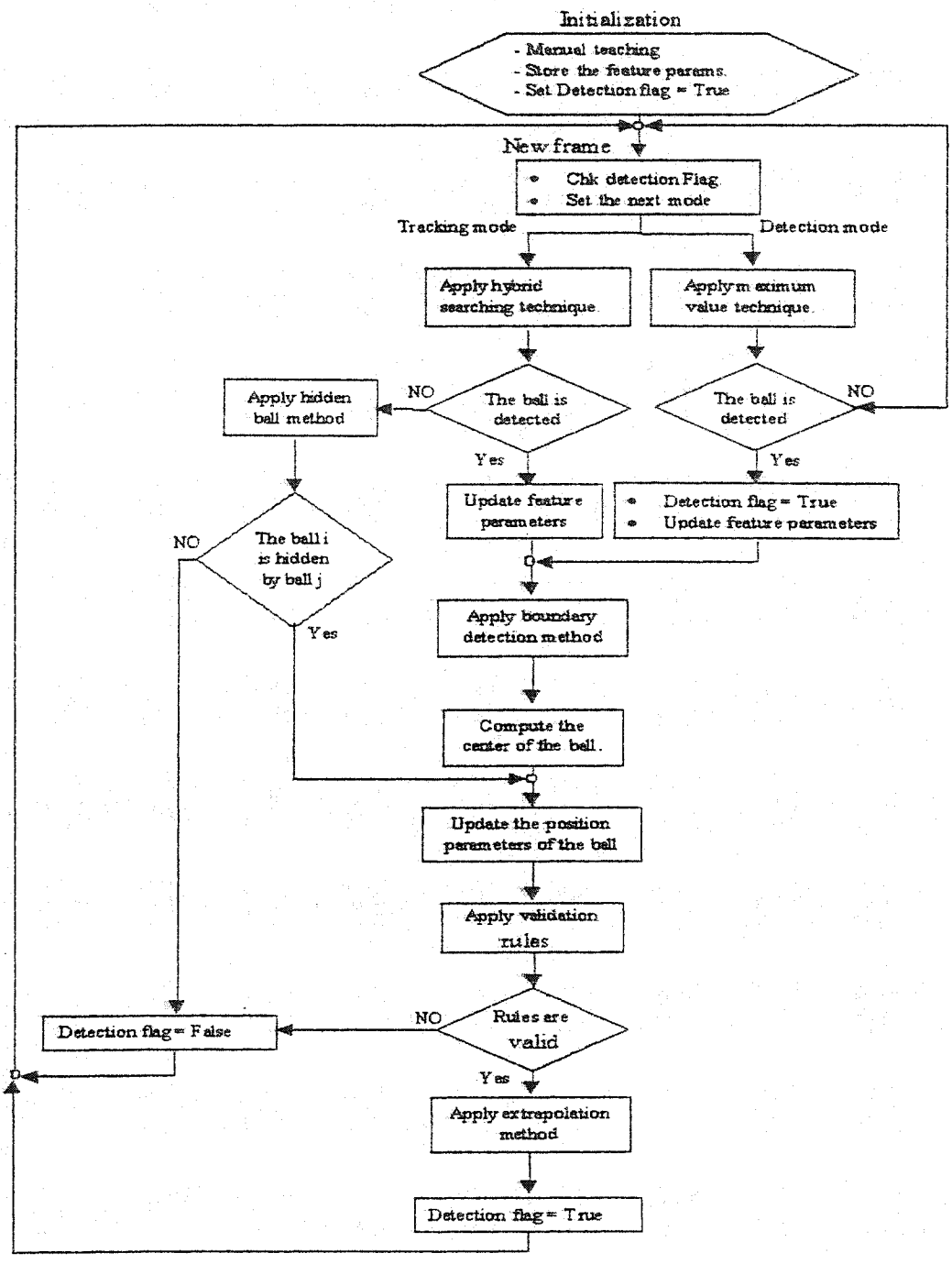

Figure 4.4: Vision program flow chart

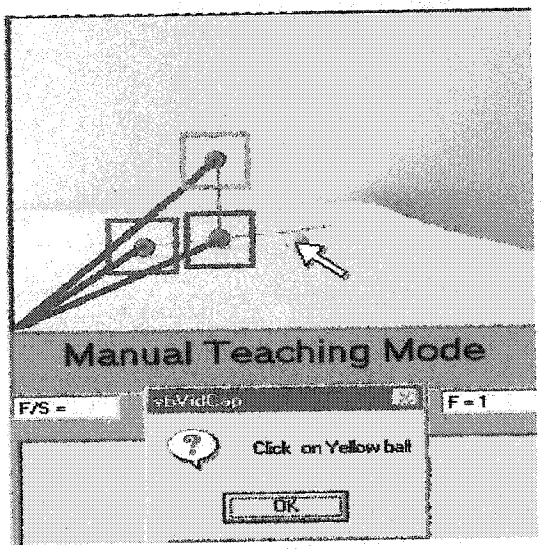

Figure 4.5: The supervisory teaching mode in the vision program

Detection Mode

This mode will be called if and only if at least one ball was not detected in the previous frame. The search area for the missing ball will be formed around the red ball or any other ball if the red ball was not detected, where the radius of the search area will be calculated from that ball. The radius is calculated by:

$$r = B_D . \frac{L}{D} \tag{4.1}$$

where r is the radius of the search area, B_D is the ball diameter in pixels as seen in the image, L and D are constants where L is the length of the link that connects the red ball to any other ball in pixels and D is the diameter of the red ball in pixels, that are measured in the initialization state. The scanning for the missing

ball will be carried on in this area by using the hybrid detection method that will be explained later in this section. The search for the missing ball will be in a bounded searching area which will speed up the detection and will minimize the risk of wrong detection. However, if the missing ball was not detected then the search area will be expanded to include the whole image.

Detection Methods

We will explain three different methods for detection:(1)**the Maximum Value Detection technique (MVD)**, (2)**the Matching Distance Detection method (MDD)**, and (3)**the Hybrid Detection method (HD)**. We will use the ball shown in Figure 4.6 for the experiments that clarifies the differences between the various methods of detection. The reader will realize the importance of a good detection method and he can find the best method that should be used for color detection from the experimental results. Every pixel on the image is represented by three bytes, which means that every primary component will be stored as a byte of information. This property of the RGB color space help us to explore the color features, and consequently detect them. The RGB values for the red ball shown in Figure 4.6 is shown in Figure 4.7.

1. **The Maximum Value Detection technique (MVD)**

   In this approach we are interested to detect the ball of color i. where $i \epsilon \{red, green, blue, yel$

   According to the information we have for colors and specifically the feature

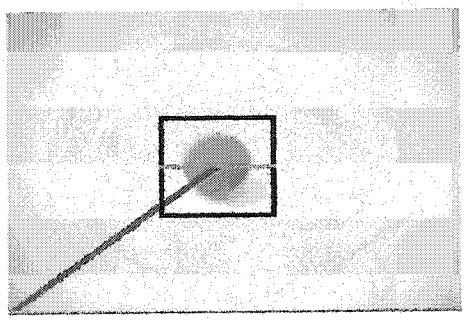

Figure 4.6: The red ball used in the experiments

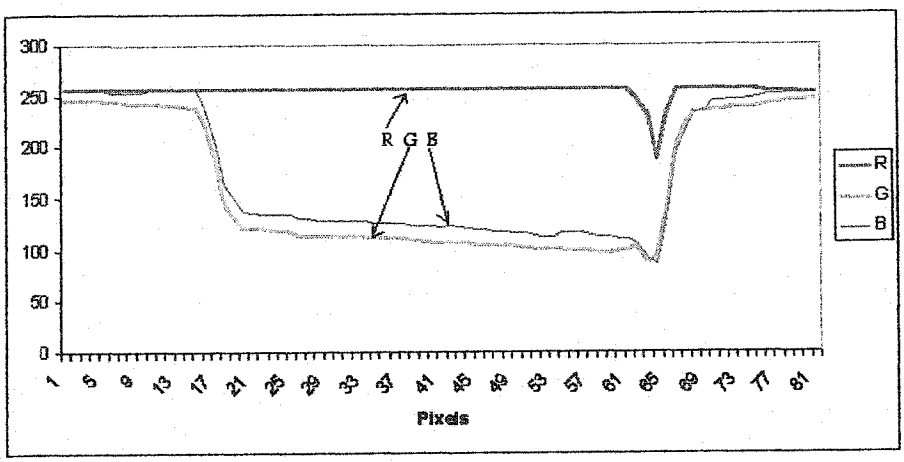

Figure 4.7: RGB components for the red ball

parameters of colors which mainly are the RGB components we formulate a formula as following:

$$MVD_c(i,j) = (256.rr - R(i,j)) + (256.gg - G(i,j)) + (256.bb - B(i,j)) \quad (4.2)$$

where the subscript c is used to indicate one of the four balls that are used in the 3D structure. R, G and B are the primary components of any color, i and j are the location of a pixel in the image frame.

rr,gg and bb have the following combination for each ball:

rr=0 , gg=1 , bb=1 for red ball

rr=1 , gg=0 , bb=1 for green ball

rr=1 , gg=1 , bb=0 for blue ball

rr=0 , gg=0 , bb=1 for yellow ball

This function gives the maximum value for the color that best match it. For example, the best combination for red color is $R = 255$, $G = 0$ and $B = 0$ which will result in $MVD_{red} = 768$. The result of the red ball used for the experiment is shown in Figure 4.8. This approach has some disadvantages described as below:

- The color that is used should be selected to be the ideal color, for example if we need to use red color we should use the ideal red color which has the following primary components: R=255, G=0 and B=0. Practi-

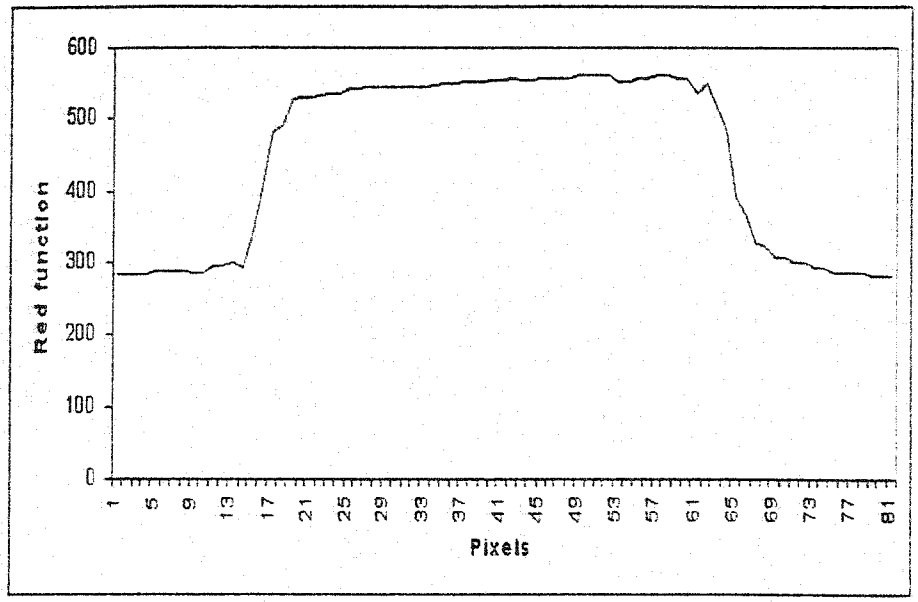

Figure 4.8: The MVD function for the red ball with white background at both sides

cal speaking this type of color selection is very difficult where the same color may give different values for its primary components under different conditions of lighting and image quality.

- There is a need for threshold values to prevent the detection if the object does not exist in the image frame. The threshold value selection is not governed by any rule and will be affected by the changes in the image and light conditions.

This shows that the maximum color approach is not very useful in color detection.

2. The Matching Distance Detection method (MDD)

In this method the distance between the actual color and the detected point

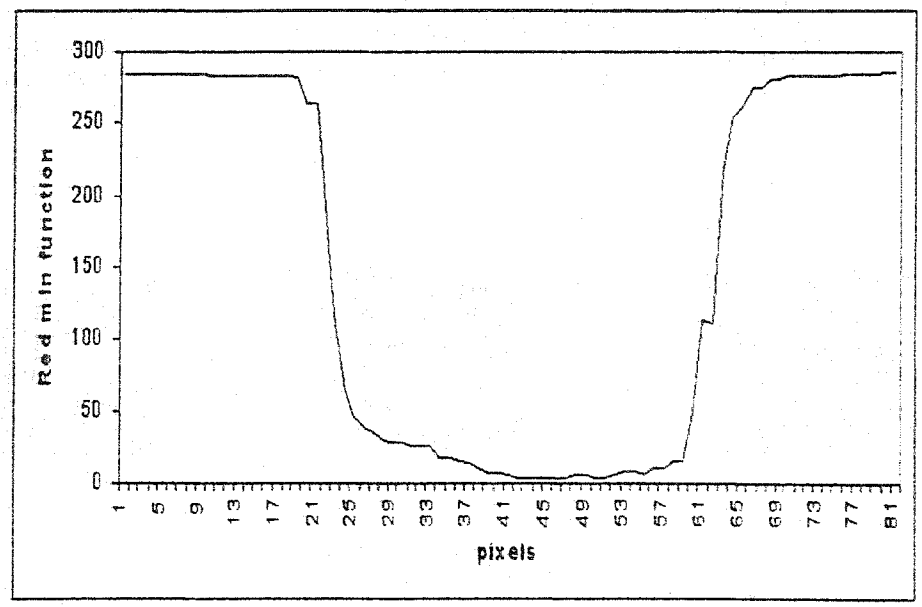

Figure 4.9: The MDD function for the red ball with white background at both sides is used for color matching. The smaller the distance between the two colors the better is the matching. The function that represent this case is given as follows:

$$MDD_c(i,j) = [(R(i,j) - R0_c)^2 + (G(i,j) - G0_c)^2 + (B(i,j) - B0_c)^2]^{1/2} \quad (4.3)$$

where the subscript c is used to indicate one of the four balls that are used in the 3D structure. R0, G0 and B0 are the feature parameters that represent a color c, i and j are the locations of a pixel in the image frame.

Figure 4.9 shows the function value for red color. This method depends on the feature parameters of a specific color to detect it. The feature parameters are collected in the supervisory mode at the beginning of the program. Although

this method of detection is very useful and gives very good results it may fail because of existing of many values that are small and approximately equal to the actual one. Then the selection of the correct object in a narrow range will be more difficult. The existence of many values that are small and close to the actual one may occur under bad lighting conditions or in noisy environment where many objects are very similar to the original one. Therefore, using this technique for color detection may sometimes give poor results. So, another approach which combine the MVD and the MDD technique, is used for color detection. The new approach is called the Hybrid Detection technique.

3. **The Hybrid Detection method (HD)**

The HD method is a combination of the MVD and the MDD functions. This approach can be descried by the following formula:

$$HD_c(i,j) = MVD_c(i,j) - MDD_c(i,j) \tag{4.4}$$

where the subscript c is used to indicate one of the four balls that are used in the 3D structure. i and j are the location of a pixel in the image frame.

This approach will overcome the need of good color selection and the threshold value problem because of using the MDD function. The range of the color detection will be wider than that of the MDD approach, because of using the MVD function which will minimize the effects of bad light conditions and noisy

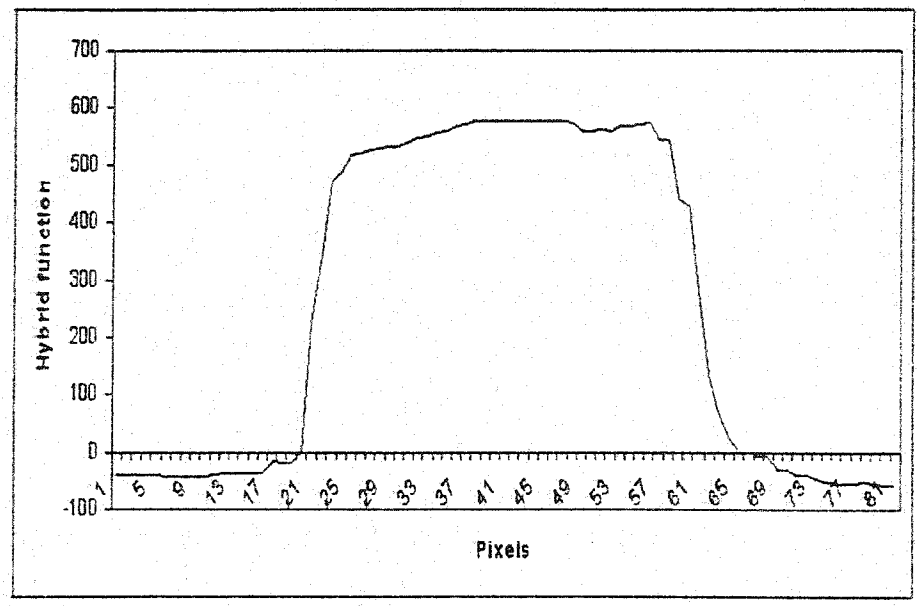

Figure 4.10: The HD function for the red ball with white background at both sides

background. Therefore, we have adopted this technique for color detection. From now on, the color detection means detection using the HD method. Figure 4.10 shows the HD function for the red color.

Tracking Mode

This mode represents the normal operation, where all the balls were detected successfully in the previous frame. In this mode, there are different approaches are used to assist tracking the ball and to measure the center and the diameter of that ball. In the following, some explanation are given for the techniques used in this mode:

The Searching Area Limits

It is very important to limit the search area and to keep it as small as possible. The search area is shown in Figure 4.11 as a small box around the ball. The reason for that is to minimize the risk of wrong color detection and to keep the image processing time as short as possible by reducing the amount of data. If we choose to detect all the points in the search area then the image processing time will increase noticeably. Therefore, we choose the number of points that are to be searched to be as small as possible subject to the ball should not be missed between points in the search area. We should mention that the search is done row-wise. The search is implemented as follows:

Fori = $X_o - nD$ to $X_o + nD$ Step D/3

  For j = $Y_o - nD$ to $Y_o + nD$ Step D/3

    Apply detection method for P(i,j)

  Next

Next

Where n is the iteration number which expand the diameter of the search area, it will start from 0 and will increase gradually until the detection occurs. X_o and Y_o are the expected position of the ball, D is the diameter of the ball in pixels, P(i,j)

is the point or the pixel at i^{th} row, j^{th} column.

In the above algorithm, we see that a number of points in each row and column are detected. These points are enough to show if the ball which is to be detected exists or not. The point that is detected represents the ball. The position of this point is usually not at the center of the ball. However, the program will call the boundary detection method in order to calculate the center of the ball. These methods will be explained later.

Hidden Ball Detection

If the ball was not detected during the detection operation, Then ball may be hidden by another ball. The distance between the balls which was measured in the previous frame will be used to verify that the ball which was not detected is hidden by another ball or not. The following algorithm is used for this purpose. If the distance between any ball and the missing ball is less than the diameter of the missing ball then we assume that the missing ball is hidden behind that ball. Otherwise, the ball could not be detected and the program will switch to the detection mode.

The Boundary Detection Method

The BD method is required to identify the edges or the boundaries of the ball in order to calculate the ball center and the diameter. The BD method is implemented by computing the HD function for each point in the background and in the ball in

a squre area containing the ball. The area height should be at least three times the diameter of the ball. Every row will be checked for the left and right edges according to the process that will be discussed below. The following algorithm is used for edge detection:

Let BKGND(i) be the HD value of the background in row i

Let HD(P) be the HD value of the detected point P at (x_p, y_p).

Let HD(i,j) be the HD value of the point at row i and column j.

For i = $y_p - 3.D\ To\ y_p + 3.D$

For j = $x_p - 3.D\ To\ x_p + 3.D$

If $HD(i,j) \geq (0.50.(HD(P) - BKGND(i)) + BKGND(i))$

Then the point at (i,j) is inside the ball

The most left point at row i is the left edge = i_L

The most right point at row i is the right edge = i_R

Else the point is not inside the ball

Next i

Next j

The boundaries of the ball can be identified as shown in Figure 4.11. We have used a threshold value, the values exceeding this value is inside the ball and the values

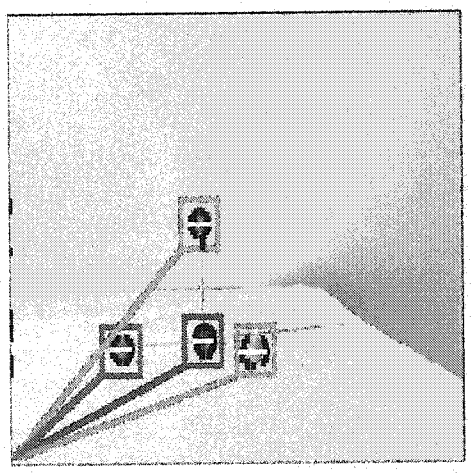

Figure 4.11: Search area, boundaries, and diameter

that are less than this value is outside the ball. The edges are considered to be the most left and the most right values inside the ball in each row cuts the ball.

The Center of the Ball Computation

Calculating the center of the ball is implemented by using the Boundary detection method. Once we know the boundary of the ball we can calculate the center of the ball and the diameter of that ball as following: Assume that X_0 and Y_0 are the coordinates of the center of the ball.

$$X_0 = i_{min} + (i_{max} - i_{min})/2$$

$$Y_0 = j_{min} + (j_{max} - j_{min})/2$$

but this method is not guaranteed to work because of these points that may be detected as a point inside the ball. For that, we choose to use a method based on taking all the points in the ball into consideration as follows:

For j = $j_{min}\ To\ j_{max}$

For i = $i_L\ To\ i_R$

$K = K + (i_R - i_L + 1)$

$N_X = N_X + (i_R - i_L + 1).i_L + ((i_R - i_L + 1).(i_R - i_L)/2)$

$N_Y = N_Y + (i_R - i_L + 1).j$

Next i

Next j

$X_0 = N_X/K$

$Y_0 = N_Y/K$

In the above program, all the pixels that lie inside the ball are included in the calculation of the center of the ball. The average of the coordinates of the pixels will be considered as the center of the ball. The diameter of the ball is computed as the largest distance between two edges of the ball. Figure 4.11 shows the computed diameter of each ball in the white line at the computed center of the ball.

Validation Rules

To verify that the data we get is correct, we use some validation rules include: (1) distance between balls, and (2) speed of the moving balls.

1. The distance validation rule:

   The distance between balls is used to validate the ball positions. The distance between balls should not exceeds the length of the link between any two balls:

$$Dist. \leq B_D . \frac{L}{D} \tag{4.5}$$

   where B_D is the diameter of the ball in pixels as seen in the image. L and D are constants and are measured in the initialization state as we mentioned before. If any ball does not pass this test then the detected point will be considered as false detection and the ball will be considered as a missing ball.

2. The speed validation rule:

   The speed of moving balls should be limited to an acceptable range of speed in order to avoid missing the balls. The speed limitation is imposed because of the limitation in the image processing time and the frame rate per second. If the speed of the moving balls exceeds a certain level then a warning message will be delivered to the operator asking him to reduce the speed of motion. The speed is measured in terms of distance per frame. Therefore, the speed

of the ball is the difference between the current position of the ball and the previous position of the same ball during a single frame of time.

The following expression is used to compute the speed:

$$s = (x(n) - x(n-1))/(t(n) - t(n-1)) \tag{4.6}$$

where x is the position of the ball, n is the frame number, and t is the frame time. If validation rules failed then we set the detection flag as false.

Extrapolation Technique

Given that the validation rules passed, then linear extrapolation technique is used to compute the future position of the ball based on the previous position and the present position of the ball. This is achieved using:

$$\hat{x}(n+1) = x(n) + (x(n) - x(n-1)) \tag{4.7}$$

4.4 Part II: Three Dimensional Computations

Here, we compute the position and orientation of the 3D object from the two images captured simultaneous by the two video cameras. six parameters are computed, three for position and three for orientation. Our main objective is to enable the human operator to move the robot arm by moving that object. The system adopts

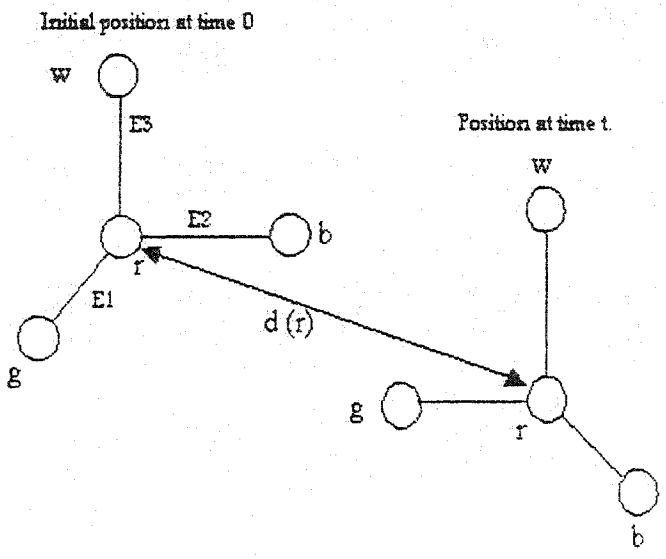

Figure 4.12: The coordinate system for the 3D object

uncalibrated stereo vision based on the affine invariant from multiple views. We have used a set of four colored points: red, green, blue and yellow. The red ball is at the origin of that structure as shown in Figure 4.1. The algorithm gives the 3D position of the object with respect to the initial coordinate system composed of the basis vectors as shown in Figure 4.12.

4.4.1 Affine Invariant from Multiple Views

We will briefly describe the multiple view affine invariance, i.e. un-calibrated stereo vision, on which our system is based. To establish a basis vector (the rods of the same length) E_i, based on the 3D object with origin X_r, which represent the red

ball, we have:

$$E_i = X_i - X_r,\ i\epsilon\{g, b, w\} \tag{4.8}$$

where the suffixes r,g,b,w represents the points: r stands for red, g stands for green, b stands for blue and w stand for yellow color. In this basis, any point can be represented by:

$$X_i = X_r + E_i,\ i\epsilon\{g, b, w\} \tag{4.9}$$

This means that we can derive two equations with three unknowns for any point location in a single 2D image. The problem is under-determined. A second view with known correspondence to the first view can, however, give an overdetermined set of equations as follows:

$$A.v = x \tag{4.10}$$

$$A = \begin{pmatrix} E_{c_1,x_g} & E_{c_1,x_b} & E_{c_1,x_w} \\ E_{c_1,y_g} & E_{c_1,y_b} & E_{c_1,y_w} \\ E_{c_2,x_g} & E_{c_2,x_b} & E_{c_2,x_w} \\ E_{c_2,y_g} & E_{c_2,y_b} & E_{c_2,y_w} \end{pmatrix} \tag{4.11}$$

$$x = \begin{pmatrix} x_{c_1,i} - x_{c_1,r} \\ y_{c_1,i} - y_{c_1,r} \\ x_{c_2,i} - x_{c_2,r} \\ y_{c_2,i} - y_{c_2,r} \end{pmatrix} \tag{4.12}$$

$$v = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{4.13}$$

where C_1 represents the first camera, and C_2 represents the second camera. The least squares solution $\hat{v}$ is given by:

$$\hat{v} = (A^T A)^{-1} A^T x \tag{4.14}$$

It shows the 3D position of the object relative to the initial position.

In addition to 3D positional information, we can obtain 3D orientation matrix R_33 information based on the 3D position information as following:

$$R_{33} = \begin{pmatrix} \frac{1}{l_1} & 0 & 0 \\ 0 & \frac{1}{l_2} & 0 \\ 0 & 0 & \frac{1}{l_3} \end{pmatrix} \cdot \begin{pmatrix} x_g - x_r & x_b - x_r & x_w - x_r \\ y_g - y_r & y_b - y_r & y_w - y_r \\ z_g - z_r & z_b - z_r & z_w - z_r \end{pmatrix} \tag{4.15}$$

We use Gram-Schmidt orthogonalization process to produce an orthogonal set of function from the set we have computed. The Gram-Schmidt process for computing an orthonormal basis $T = \{Z_1, Z_2, .., Z_m\}$ for a m dimensional subspace W of R^n with basis $S = \{X_1, X_2, ..., X_m\}$ is as follows.

Step1. Let $Y_1 = X_1$.

Step2. Compute the vectors $Y_2, Y_3, ...Y_m$ successively, one at a time, by the formula

$$Y_i = X_i - (\frac{X_i.Y_1}{Y_1.Y_1}).Y_1 - (\frac{X_i.Y_2}{Y_2.Y_2}).Y_2 - ... - (\frac{X_i.Y_{i-1}}{Y_{i-1}.Y_{i-1}}).Y_{i-1}$$

The set of vectors $T^* = \{Y_1, Y_2, ...Y_m\}$ is an orthogonal set.

Step3. Let

$$Z_i = \frac{Y_i}{\|Y_i\|}, 1 \leq i \leq m$$

Then $T = \{Z_1, Z_2, ..., Z_m\}$ is an orthonormal basis for W.

4.5 Part III: Experimental Human-Robot Interface System

The system configuration that is used for the experiments is shown in Figure 4.13. The system consists of two personal computers, connected to each other via a serial port. Each computer is equipped with a video card. One computer acts as a client and the other one acts as a server. The client PC will extract the features of the image captured by the video card and will send the information to the server PC. The server PC will receive the information of one image from the client and it will use this information together with the information recovered from the other camera to compute the six parameters for the object. The personal computer transmits motion commands to the robot according to the result of the object motion. The robot arm

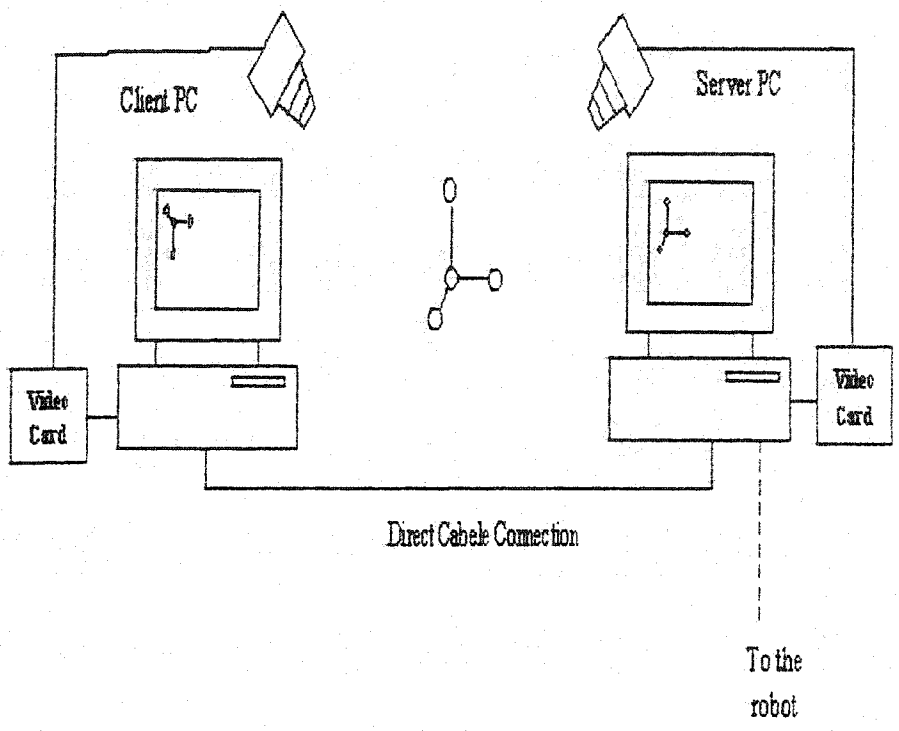

Figure 4.13: Experimental system

is equipped with a camera, whose image is sent to the server PC. This will enable the operator to work in a closed loop with vision feedback. The experimental results show the usefulness of the system Our system can track the object and calculate the 3D information at 9Hz, where we use personal computers of 350MHZ speed and 128MB RAM. The main reason for this time delay is the time spent for computation of the ball centers and the image input overhead. The performance of the system can be improved easily by using faster PC components/hardware.

Chapter 5

TELEROBOTICS

Telerobotics can be defined as a remote control of robots by a human operator. Most traditional telerobotics have closed loop control with vision feedback, force feedback or both. The operator closes the loop and control the robot arm by using a master arm while he is receiving the information from the robot side. The main components in the telerobotics are the human operator, the master arm, the slave arm, the vision feedback, and the communication link. We have designed an intelligent application programming interface that incorporates all the above components of the telerobotics in one module. The module is called the client-server system. This module is described in Section 5.1.

The control of the robot arm is another main issue in the telerobotics. Since the master arm configuration is different from the robot arm configuration. We need a good mapping between them to maintain good control. We introduce this issue of

control and mapping in Section 5.3.

5.1 The Client-Server System

A robust client-server application is essential for successful operation of a telerobotics system. The client-server application is the medium that facilitates the data interchange between the master arm and the slave arm. The client-server application is composed of two separated environments: the client environment and the server environment. These two environments are connected to each other over a LAN of 100Mbps bandwidth. The client environment is composed of a master arm, a personal computer (PC), and a human operator. The server environment consists of a robot (slave) arm, video cameras, and a personal computer (PC). Figure 5.1 shows the components of the system. We call the PC at the client side as the client-PC, and the PC at the server side as the server-PC. In the following, both environments will be explained in detail.

5.1.1 The Client Environment

In the client side there are three major elements: (1) the master arm, (2) the human operator, and (3) the client-PC.

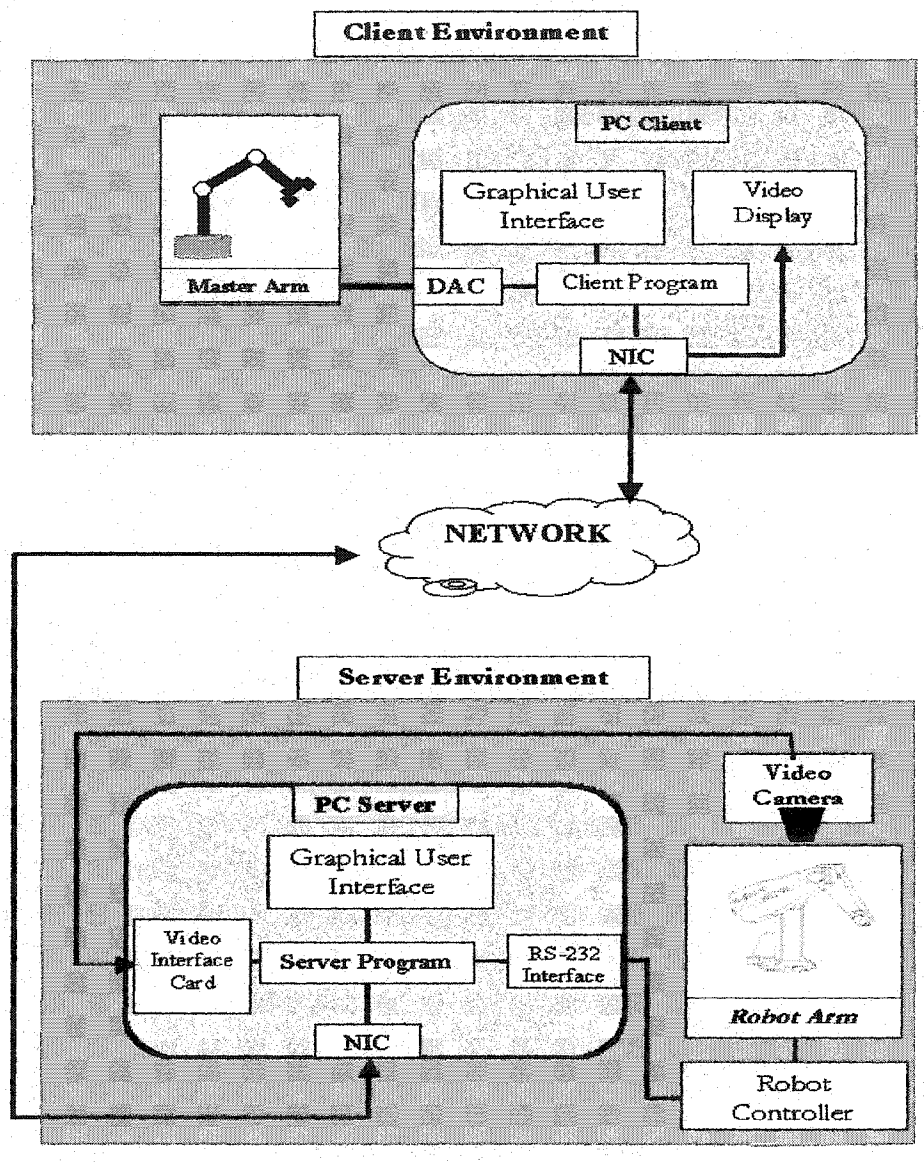

Figure 5.1: The Client-Server system

The Master Arm

The master arm consists of six links with six revolute joints. Each joint is equipped with a sensor which is used to measure the angle between joints. In addition to these six sensors, there is a Stop/Start button attached to the master arm holder at the end part of the master arm. This button is used to enable/disable the master arm, and hence can be used by the operator to disable the system temporarily, move the master arm to a comfortable location, and then enable the system. The master arm is connected to the client-PC by means of a data acquisition card. The data acquisition card collects the data from the sensors attached to the master arm joints. The data collected by the card will be digitized by an Analog-to-Digital Converter (ADC). Analog signals that are delivered by the sensors are collected through seven different I/O ports. These signals are in the range 0 to 10 volts. The data acquisition card is accessed via the Ntport library freely available from Zeal SoftStudio company. The master arm is shown in Figure 5.6.

The Client-PC

The client-PC is the programming environment that is used to interconnect all the elements in the client environment with each other. Also the client-PC is used to collect data from the master arm and to transform it from the joint space into the Cartesian space by using the geometric model of the master arm which was explained in Chapter 3. The client-PC is connected to the server-PC over the LAN. The data

interchange between the two sides is based on request from one side to the other. This type of data flow can be considered as a Stop-and-Wait flow control, and is implemented because of the nature of the robot motion and control. In fact the robot arm will not accept any request until it finishes its current motion. The data interchange between both sides includes information from the server about the robot and the server status. Also the client-PC will receive a stream of real time video showing the robot arm. The remote user will receive both textual and graphical information about the current state of the robot and the server. The outgoing data from the client-PC to the server-PC will include information about the client status, and the incremental motion of the master arm in the Cartesian space. An informative, yet simple interface is essential for teleoperation to be easy for use and productive. The client-PC interface is shown in Figure 5.2. The complexity of kinematics, work space boundaries and data manipulation are hidden from the operator.

5.1.2 The Server Environment

In the server side there are also three major elements:

1. The slave arm (The robot arm),
2. Video cameras,
3. The Server-PC.

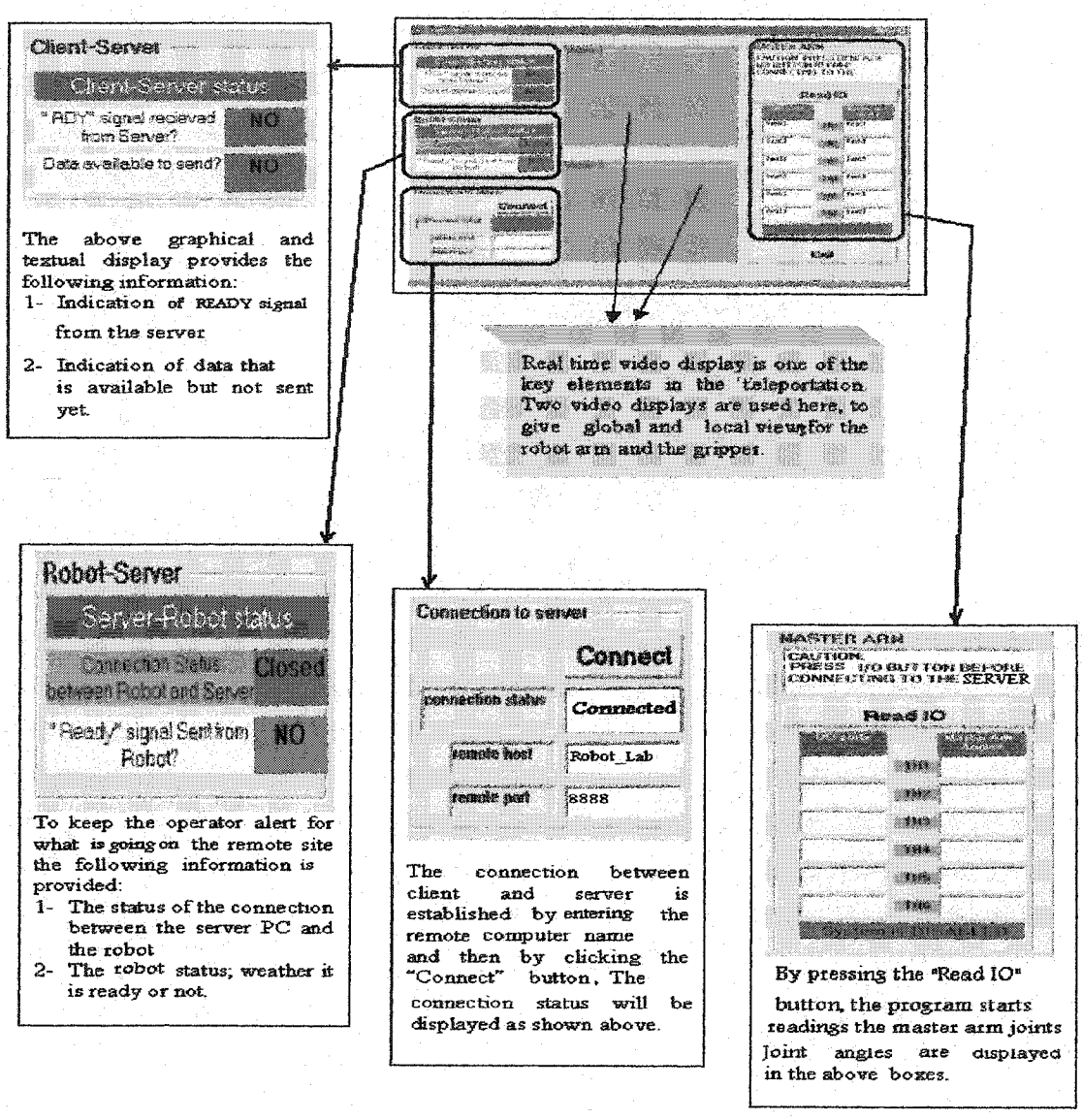

Figure 5.2: The Client-PC Interface

The Slave Arm (PUMA-560)

The PUMA-560 system consists of two basic units: The robot arm and the controller. The robot arm has 6 degrees of freedom with all revolute joints. The arm is connected to the robot controller, which in turn connected to the server-PC. The controller is connected to the server-PC through a serial communication link (Via the RS-232 interface). The robot software that controls the robot arm is called VAL and is stored in the memory of the control module. There are a list of instructions for the action to be taken, and a list of locations at which these actions are to be taken. The controller will receive the instructions from the server-PC and will generates command signals for robot arm. Position data obtained from incremental encoders and potentiometers in the robot arm are transmitted back to the controller and then to the slave arm to provide closed loop control of the robot arm. If the actual position of a joint measured by the encoder is different from the position commanded by the operator by more than a preset error value, then this joint will force VAL to declare a FATAL ERROR condition. This error is called Position Envelope Error, and will cause shut-down of Arm power.

The Video Cameras

One of the essential elements in a teleoperation system is the real time video. Two cameras are used to take real time video for transmission to the client-PC. We have used two well-known applications to transfer these video streams from the server-PC

to the client-PC, the MSnetmeeting and the CuSeeMe application.

The Server-PC

The server-PC is at the center of the teleoperation system. It is connected to the client-PC over the LAN to the robot controller by RS-232 interface and is connected by video cameras. Therfore, the server-PC is like the brain of the system that receives data and instructions from different parts in the system, processes them, and sends appropriate commands. The server-PC interface is shown in the Figure 5.3. The first thing the server-PC does is to instruct the robot to move to a pre-defined location. Then the server-PC uploads a small routine to the robot controller. This routine or program resides in the controller memory and will wait for location parameters from the server-PC. The position data obtained from incremental encoders and potentiometers in the robot arm are transmitted back to the controller and then to the slave arm to provide closed loop control of the robot arm. The server-PC will send the new location parameters to the robot controller once it receives the position data and a ready signal from the robot. The new location parameters are computed by adding the incremental motions performed by the master arm to the actual position parameters of the robot arm in the Cartesian space. Then the computed position vector and orientation matrix can be converted to a joint space by using the inverse geometric model of the robot arm. Only the incremental motion will be sent to the robot arm to perform the desired motion.

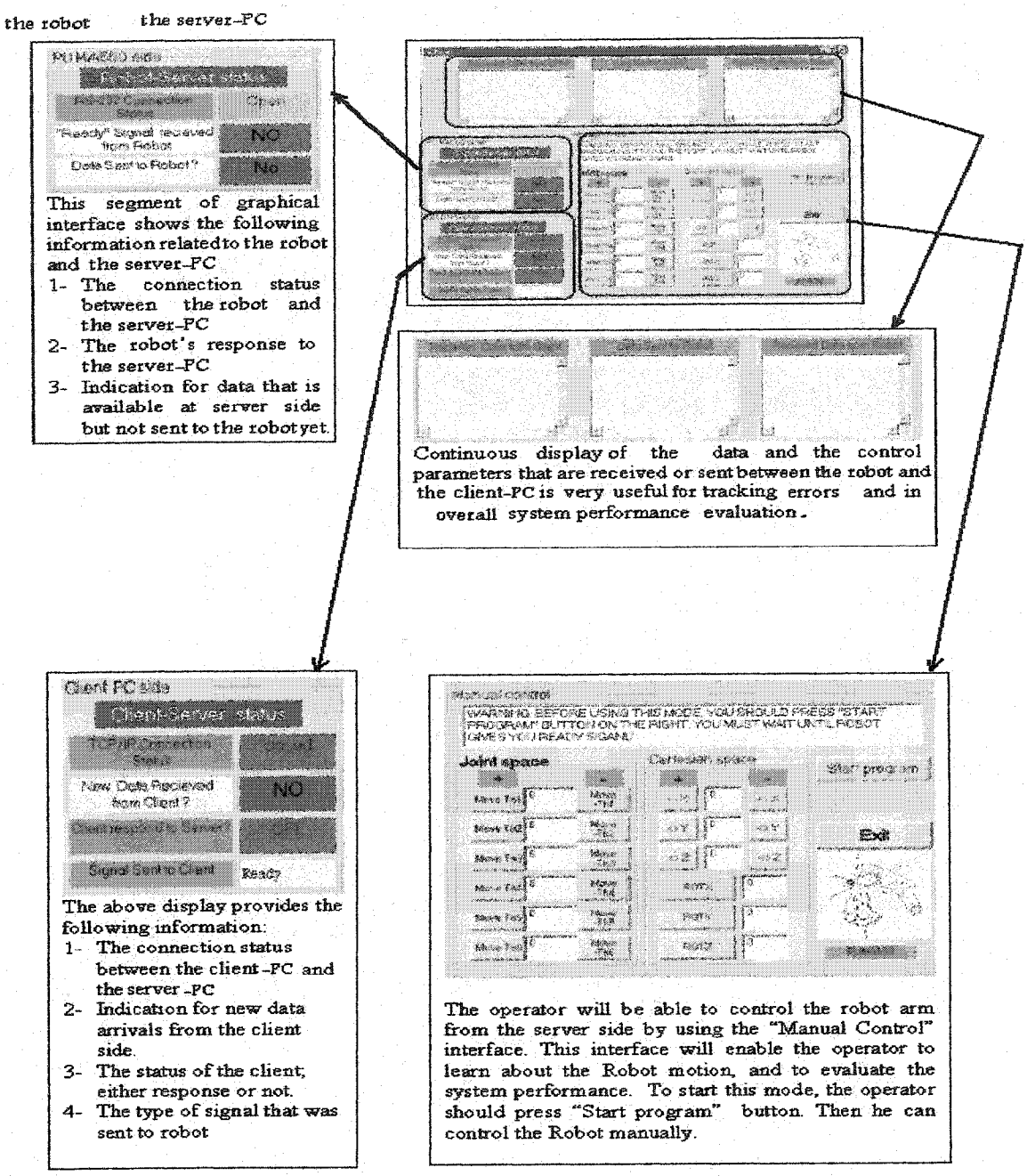

Figure 5.3: The Server-PC Interface

5.1.3 The Client-Server Interaction

In this section we will explain the interaction between the different parts of the tele-operation system. The data flow can be divided into two parts: The Load/Initialize step and the active system state. The Load/Initialize flow diagram is shown in Figure 5.4. In this state, the program will start at the client side and will establish connection with the server. Then a robot control program that directly control the robot will be uploaded to the robot controller. The robot control program is a VAL-II program that is written to accept the incremental motion parameters performed by the master arm. Then the robot control program will cause the robot arm to move accordingly, starting from its current position. After the robot control program is uploaded to the robot controller successfully, it will send a ready signal to the server indicating that it is ready to accept the new motion parameters.

The next state is the normal operation state or what we call the active system state. The flow diagram for that state is shown in Figure 5.5. During this state the server will receive data and control signals from the client and from the robot sides. Initially the server will receive ready signal from the robot, then it will request the client to send a new data if available. The data that is received from the client will be processed at the server side and will be sent to the robot arm as the new motion parameters. There is some possibility for data loss and errors. Therefore, we use timers as per a Stop-and-Wait flow control with possibility of errors, to recover the

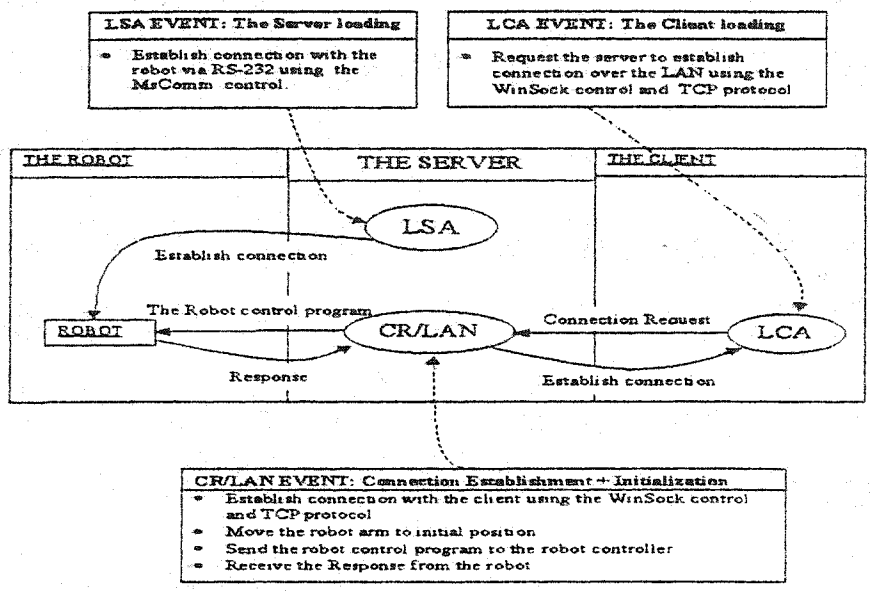

Figure 5.4: The Load/Initialize step

system from these states.

The Load/Initialize Step

The flow diagram shown in Figure 5.4 explains the flow of data and control signals between the client, the server and the robot at the loading time of the program. The flow diagram is divided into three parts. The middle part represents the server-PC. The server-PC is the backbone of this system, where it should control the flow of all data and signals between the client-PC and the robot. This step is considered as a transition state. After loading and initialization, the active system state will be activated. The client-PC is shown at the right side of the diagram, while the robot is at the left part of the diagram.

Active System State

The active system state is shown in Figure 5.5. This state is active during the normal operation of the system. It is also divided into three parts. The middle part represents the server-PC. This part is as we mentioned before the backbone of the system. The server-PC receives data from the client and process them, then it sends them to the robot. Also the server-PC receives data from the robot and process them, then it request the client to send new data if available. In addition to the data flow control, the server-PC will take care of the control signals flow in all directions. The other two components of the system are the robot and the client-PC. They are shown at the left and right side of the flow diagram respectively.

5.2 Teleoperation with Master Arm

Teleoperation and telerobotics are technologies that enables remote operations. Many teleoperation and telerobotic systems use dedicated communication links between human operator (master) and the remote manipulator (slave). Recently the LANs and Internet has supplied the communication link for many systems. In some cases, teleoperation systems include force feedback. Teleoperation technology supports a form of control in which the human directly guides and causes each increment of motion to be applied to the slave. Typically the slave robot follows the human motion exactly although in more advanced, computer controlled, sys-

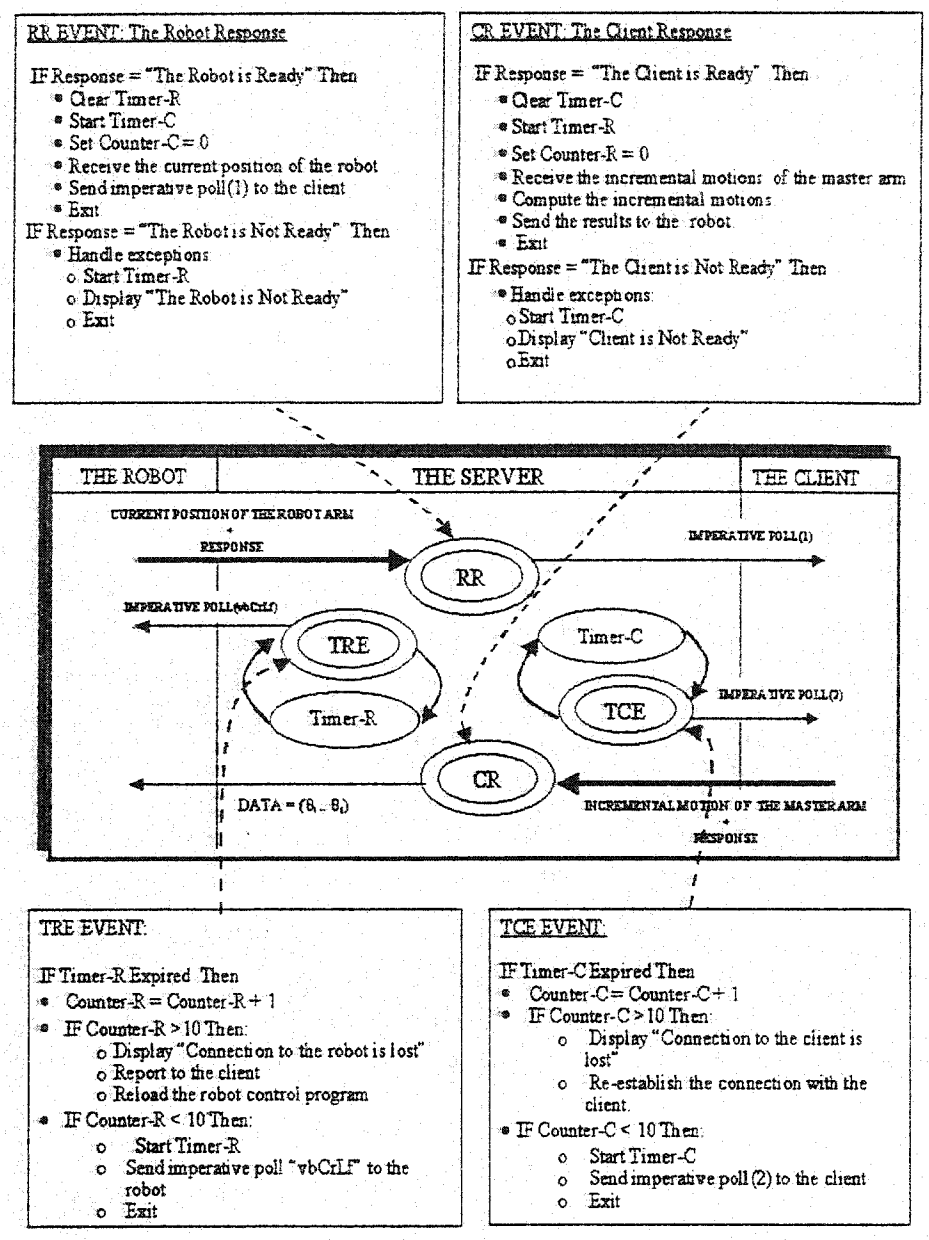

Figure 5.5: The Active system state

tems there may be coordinate transformations imposed between the master and the slave. Many systems that are remotely controlled allow the operation to one of two predefined states, either on or off. Thus, we say that the remote control system is not a telerobot if it only permits the on/off selection of state. In many systems, master and slave arms are geometrically identical within a scale constant. In this case it is sufficient to transmit the angles of the individual joints in the master arm. Because of the geometric similarity, the motion of the slave end effector will exactly follow that of the master arm. Both master and slave side motions are therefore interconnected in joint coordinates. The original teleoperators developed for the nuclear operations used joint coordinates interconnection. For many applications, it becomes desirable for the master arm and slave sides to be geometrically different. In this case, the joint coordinate of the master arm do not specify the desired joint motion of the slave. Coordinate transformations based on the kinematic equations of the two device are required to resolve these different languages. This coordinate transformation became feasible in real time, because of the rapid advance in computer technology. For example, our master arm is designed independent of the slave arm, by completely different people/companies. Such systems where master and slave have different geometric designs are called generalized teleoperation systems. Our passive master arm is shown in Figure 5.6, and is used to remotely control the PUMA-560 robot arm. The communication link between the operator and the robot is the LAN of 100Mbps bandwidth. The master arm has six serial degrees

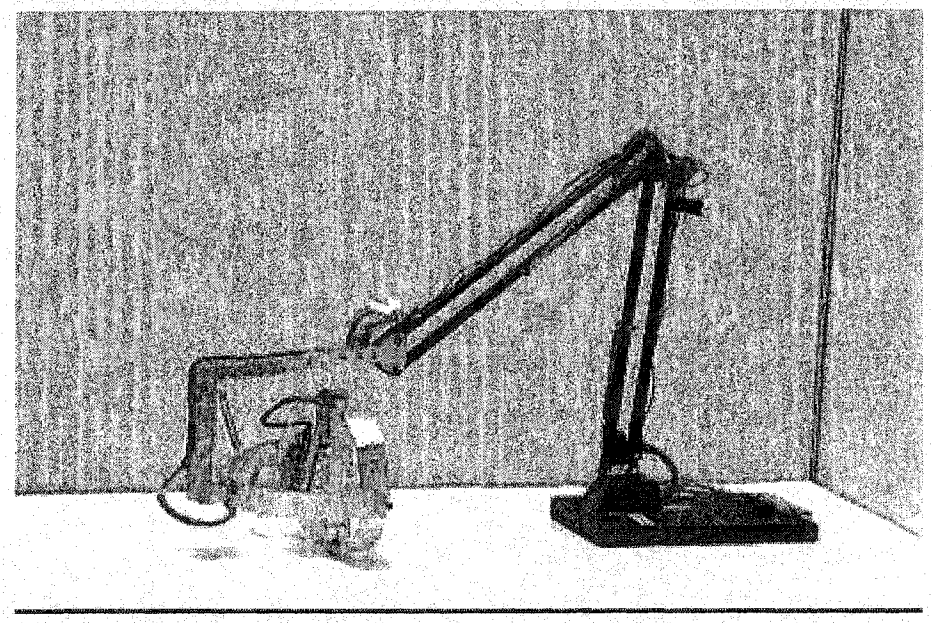

Figure 5.6: The final shape of the master arm

of freedom with all revolute joints. Each joint is equipped with a potentiometer (sensor) which are used to measure the angles between joints. In addition to these six sensors, the holder at the end effector part of the master arm is attached with a Stop/Start switch and mode selector switch. The stop/start button is used to enable/disable the master arm, and hence can be used by the operator to disable the system temporarily, move the master arm to a comfortable location, and then enable the system. The mode selector switch is used to select the mode of operation. A data acquisition card is used for interfacing the master arm with the personal computer. A visual basic environment is developed to allow the communication between master arm and slave arm. The slave arm(PUMA-56O) is connected to the server environment, while the master arm is connected to the client environment.

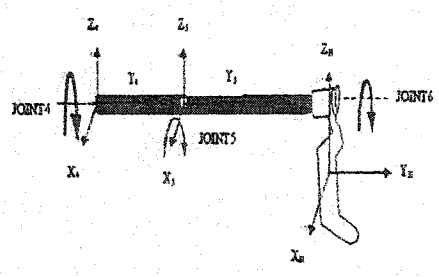

Figure 5.7: The first end-effector design

The Effector Part of the Master Arm

The effector part of the master arm is the orienting machine which is defined by the three revolute joints. θ_4, θ_5 and θ_6 with three concurrent rotation axes. The design of the effector part was changed several times for improvement and development purposes, while the transporter part remains the same. The effector part shown in Figure 5.7 was the first design. We found later that the construction of this part is not suitable for the remote control operation, for the following reasons:

- A singularity problem arises when the joint 5 becomes along Y_4 axis. In that case the joint 5 cannot rotate about Z_5 axis and as a result the operator cannot rotate his hand about Z_H axis.

- The mechanical impedance is not identical in all directions which means that the operator cannot make abstraction of the structure due to lack of symmetric impedance.

- When the operator rotates his hand about Z_H, Y_H, or X_H, the transporter part

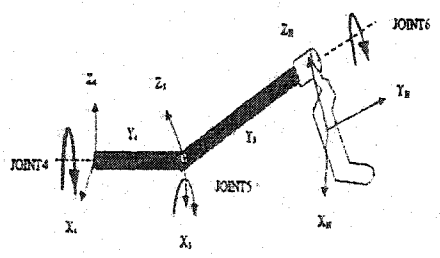

Figure 5.8: The effector part with a small angle for joint 5

will be translated for some distance which illustrates the mechanical coupling problem between the effector part and the transporter part.

To avoid the problem of singularity we decided to operate with a small angle for joint 5 as shown in Figure 5.8. This approach of operation partially solve the problem of singularity, but the operator hand get away from the origin of the concurrent rotations. To keep the operator hand at the origin of the concurrent rotations, we modified the effector part as shown in Figure 5.9 which partially solved the problems of singularity and the origin of rotation. Although this effector part gave better results and allow better operation than the previous one, it could not overcome the mechanical coupling and the non-symmetric impedance problems. We decided to build another effector part as shown in Figure 5.10, which overcomes all the problems of the previous structures. This effector part enables the human operator to make abstraction of the master arm and concentrate on the operation and the control of the robot arm.

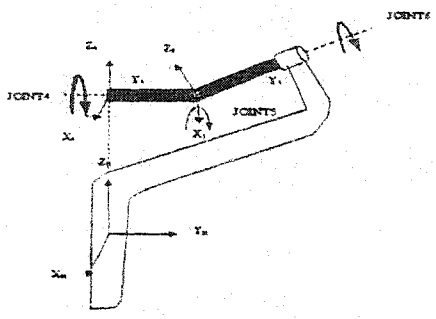

Figure 5.9: The modified effector part

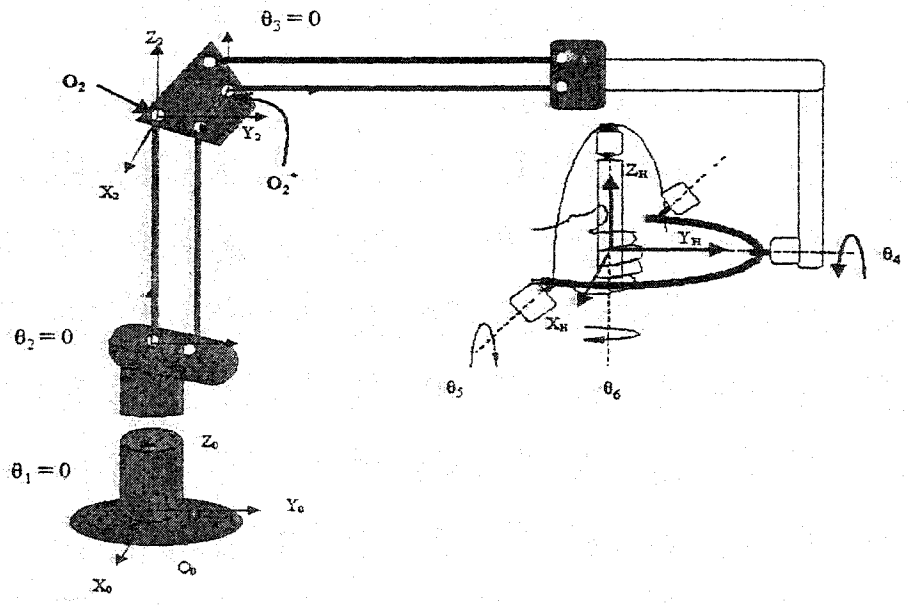

Figure 5.10: The final design of the effector part

5.3 Mappings

To solve the problem of the difference in the geometric designs between the master arm and the slave arm (PUMA-560), we developed two mapping schemes. In which the operator at remote side can control the robot arm at the other end of the LAN by using the master arm. Initially we have developed the first mapping scheme. This scheme allows us for the first time to remotely control the robot arm. However, there are some drawbacks which complicate the control of the robot arm. Consequently, we developed the second mapping scheme. In the second mapping scheme we got excellent results. In the following, we will explain both mappings schemes.

5.3.1 The First Mapping Scheme

The first scheme was developed based on the idea that the operator will virtually operate the end point of the slave arm, while he is holding the end point of the master arm. In this scheme the position and orientation of master arm is assigned to the slave arm. The operator does not know the actual position of the slave arm, instead the master arm vector E_m is used to find the corresponding slave angles θ_P by using the inverse geometric model of the robot arm. All the computations of the control system are implemented in the client side. First of all, the angles of the

master arm are gathered via the data acquisition card and then transformed to the corresponding position X_m and orientation matrix Φ_m, by using the direct geometric model of the master arm, $E_m = G_m(\theta_m)$ where $E_m = (X_m, \Phi_m)$. The vector E_m is used to find the corresponding slave angles by using the inverse geometric model of the slave arm as following:

$$\theta_p = G_p^{-1}(E_m) \tag{5.1}$$

The computed angles are incrementally assigned to the slave arm so that the end points and orientations of the slave and the master arm are equal. This scheme requires the configuration of both arms to be initialized to the same position. This is based on one single configuration mapping, but any shift will result in loss of correspondence. And as a result, the quality of operation will degrade significantly. This also means that there is accumulation of errors. The dexterous operation is not possible in this scheme, because shifting the master arm will lead to losing the control of the robot arm. The control system block diagram of the first mapping scheme is shown in Figure 5.11. The only feedback signal the operator receives from the robot side is the image data. This data is not enough to control the robot arm in an efficient way. Therefore, with this control scheme there is no way to correct the drift between the master arm and the slave arm.

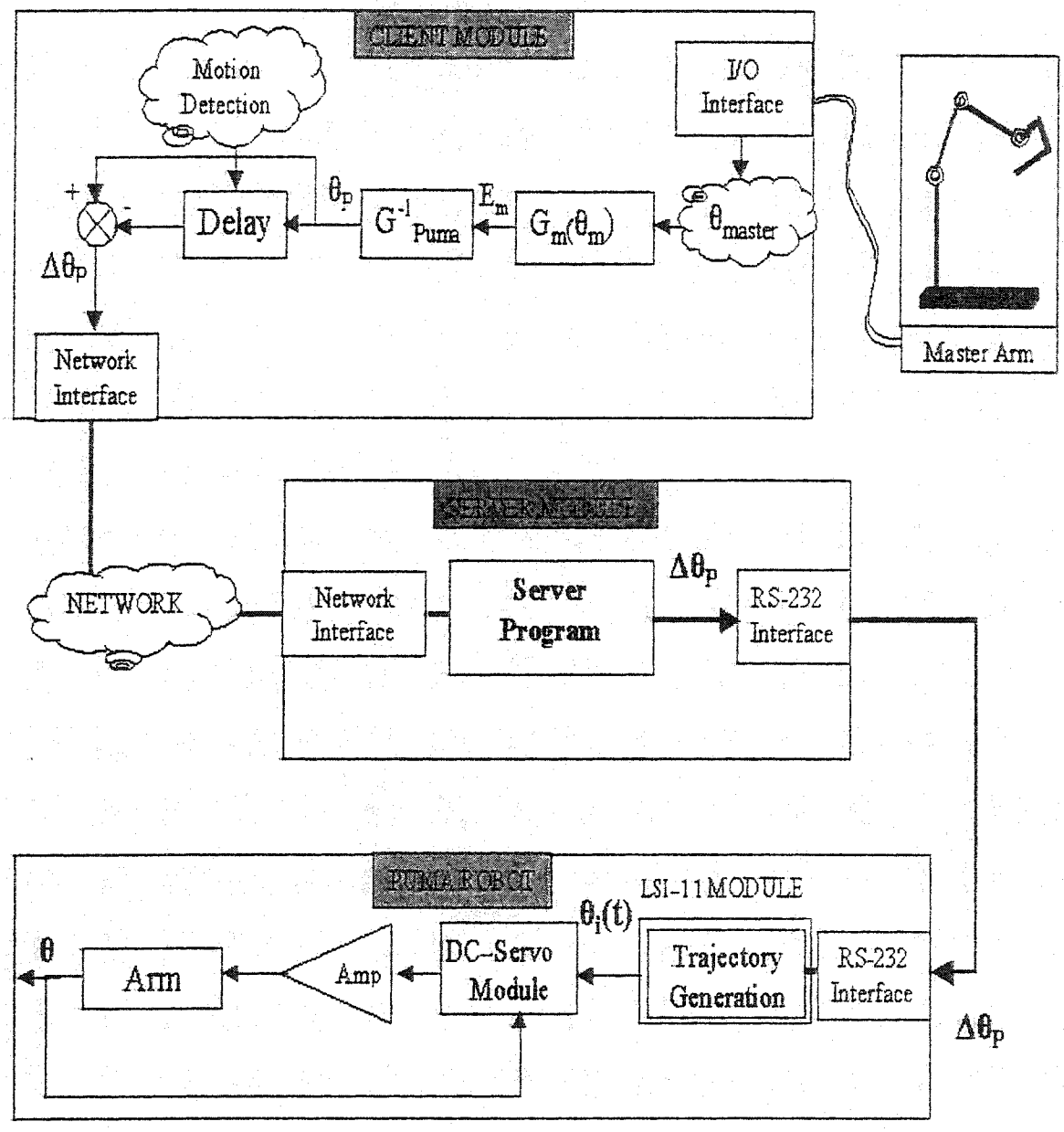

Figure 5.11: The first mapping scheme.

Evaluation of the First Mapping Scheme

As we mentioned above, once the mapping is lost, the system get out of control and can not be returned to the correct state. There are few reasons for the loss of mapping:

1. Loss of data: This may happen at any time during the operation where the operator moves the master arm to some position but the data that is sent is lost. The data may get lost or deformed at client side, network or at server module.

2. Different solution: There is a chance that the robot arm receives a solution that is not acceptable, because of the different configuration of the master arm and the robot arm. In this case the mapping will be lost once it receives a wrong solution.

We experimented with this mapping scheme and we found that it needs very large number of trials every time we need to set up the robot arm at the same position and orientation. We found that this mapping scheme is poor and is not practical.

5.3.2 The Second Mapping Scheme

In the second mapping scheme, as shown in Figure 5.12. The changes in the operator effector ΔE_m is assigned to the current effector of the slave arm, regardless of the current position of the operator hand. The computation of the master arm positions

will be implemented in the client side. The increments in the motion of the master arm $\Delta E_n^m = (\Delta X_n, \Phi_n)$ Where:

$$\Delta E_n^m = G_m(\theta_n^m) - G_m(\theta_{n-1}^m) \tag{5.2}$$

$$\Delta X_n^m = X_n^m - X_{n-1}^m \tag{5.3}$$

$$\Phi_n^m = M_{n-1}^{T^m}.M_n^m \tag{5.4}$$

where n represents the current value, (n-1) represents the previous value and m is the symbol for the master arm.

The motion increments will be sent to the server module, where they will be added to the actual position parameters of the slave arm. The result of this operation is the new position parameters:

$$X_{n+1} = X_n^p + \Delta X_n^m \tag{5.5}$$

$$M_{n+1} = M_n^P.\Phi_n^m \tag{5.6}$$

where (n+1) represents the new or the expected value, and p is the symbol for the PUMA-560 robot arm.

The new position parameters , X_{n+1} and M_{n+1} will be transformed to the corre-

sponding slave angles by using the inverse geometric model of the slave arm(PUMA-560 Arm):

$\theta_{n+1} = G_p^{-1}(E_{n+1})$ where $E_{n+1} = \{X_{n+1}, M_{n+1}\}$.

The new slave arm angles will be subtracted from the current angles of the robot arm as following:

$$\Delta\theta = \theta_{n+1} - \theta_n^P \tag{5.7}$$

$\Delta\theta$ will be sent to the robot controller and there they will be added to the current robot angles and will cause the robot motion. The actual position data of PUMA-560 are obtained from incremental encoders and potentiometers in the robot arm. This feedback signal from the robot arm provides a closed loop control of arm motion.The advantage of this control system is that the operator forgets about the master arm and only thinks that he is mapped to the gripper frame, of the robot arm without any consideration to the master arm position or configuration. This mapping scheme is dexterous, because the master arm can be shifted without affecting the mapping. This type of mapping between the master arm and the slave arm is dynamic and ensures there is no accumulation of errors, because the corrections are always made to the current robot effector.

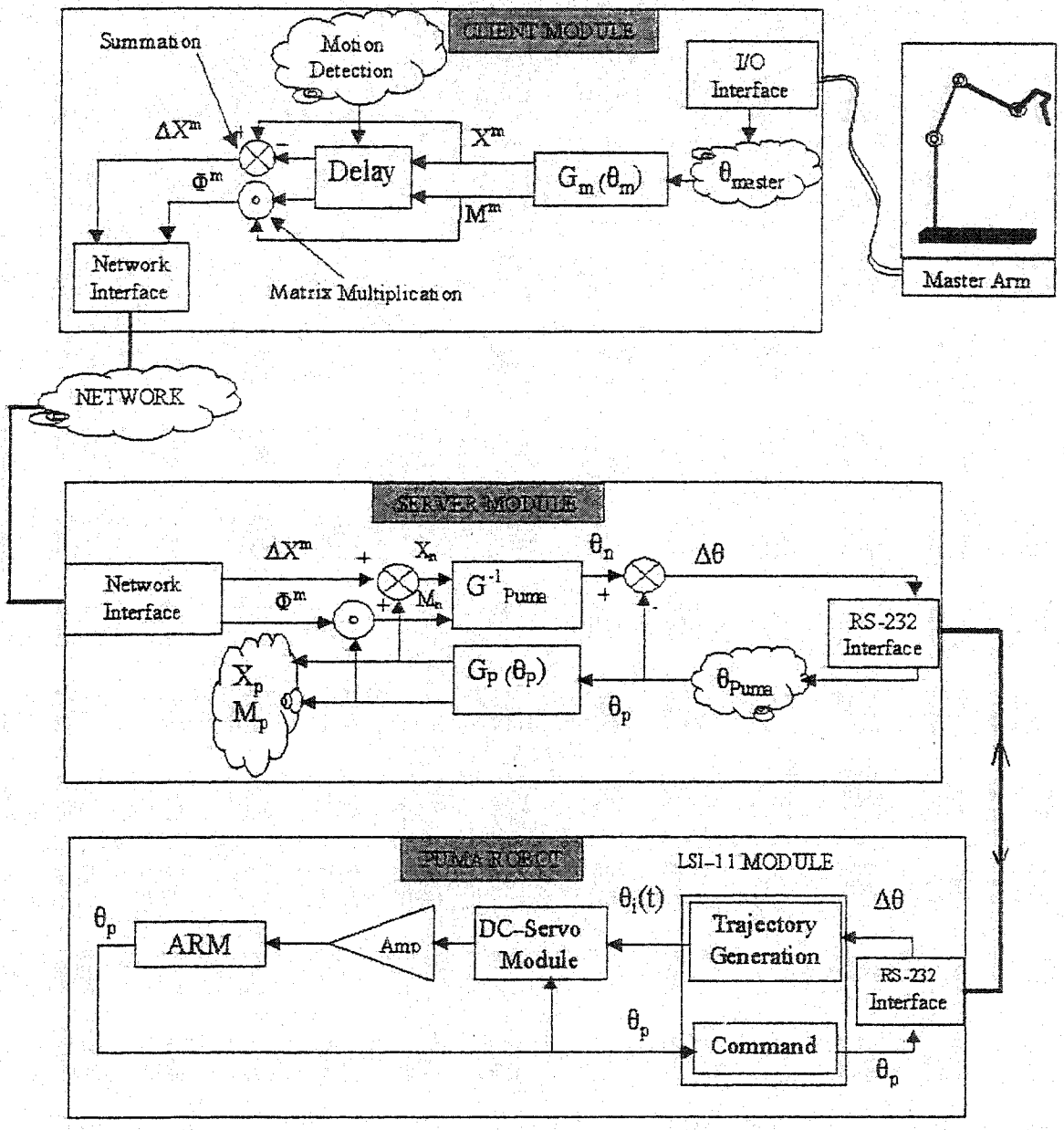

Figure 5.12: The second mapping scheme.

Evaluation of the Second Mapping Scheme

We have developed a mathematical interface at both client and server side. The interface is used to test the second mapping scheme before applying it to the master arm. The mathematical interface as shown in Figure 5.13 accepts the keyboard input.

We found this mapping scheme to be robust and universal. In the following we will give some explanation for the advantages of this mapping:

- Universal input: Once the incremental position and orientation is supplied to this mapping scheme, it will do the motion precisely regardless of the data source.
- Master-Independent: This mapping will accept the incremental motion from any master arm regardless of its configuration.
- Language-Independent.
- Scalable : The incremental position can be scaled up or down to fit the user requirements.

Anyhow, controlling the robot arm to perform some task is not an easy job. It requires some practice, specially if you are using the base frame as your reference. We will explain the different frames that are attached to the robot arm in the next section.

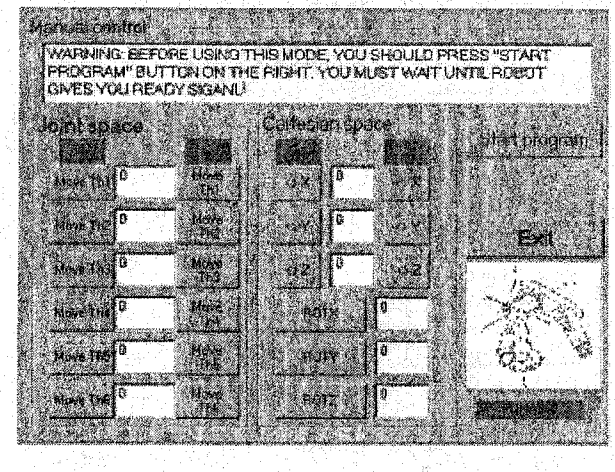

Figure 5.13: The mathematical interface.

5.3.3 Mode Selection

We have classified the modes to match the frames attached to the robot arm. The three frames shown in Figure 5.14 are used as a reference to the modes that are used. All robot motions will be described in terms of these frames. The operator selects between the different modes, to perform his task easily. This facility makes it easier for the operator to control the robot arm in a more intelligent way. For example, he may find the operation with the base frame mode easier if he likes to perform a straight motion. If the operator likes the tool to move forward and backward without any change in the orientation, he may need to use the tool frame mode...etc.

Brief explanation of the modes and their corresponding frames shown in Figure 5.14 are listed below :

1. The base frame mode:

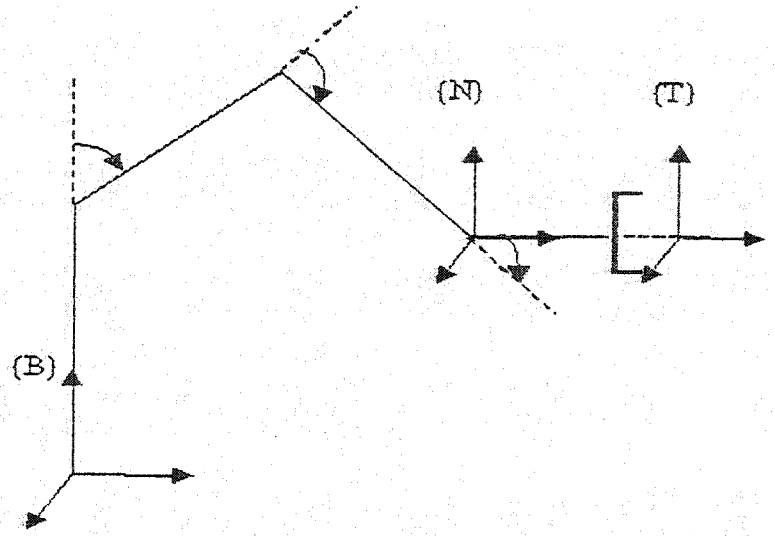

Figure 5.14: The PUMA-560 frames.

The base frame {B} is located at the base of the robot or at some distance from the base. The base frame is also known as R_0. When the operator selects this mode, the motion of the robot will be described in terms of the world frame. In this mode the position and orientation of the tool frame which is attached to the end-effector is described or calculated relative to the base frame {B}.

2. The wrist frame mode:

   The wrist frame {N} is the frame attached to the last link of the transporter part of the robot arm (link 3). The origin of this frame is fixed at the wrist of the arm, and moves with the link 3. In this mode the position and orientation of the tool frame is described or calculated relative to the wrist frame {N}

3. The tool frame mode:

The tool frame {T} is fixed to the end of any tool the robot is holding. In this mode the position and orientation of the tool frame is described or calculated relative to the frame attached to itself.

With the proper mode selection and by using good mapping scheme, the control of the robot arm becomes easier and better. with excellent systems the operator concentration is directed to perform his task as he is using his hand without any regard to the master arm configuration. The human operator will be asked to enter inputs in the form of keystrokes and mouse clicks to remotely operate and control the motion of the robot arm. How natural his interaction with the robot arm? How easy is it to drive the robot arm to a specific location?

The answer will clarify that the remote operation by using the mouse or the keyboard will not enable the operator to interacts naturally and he can not forget that he is using these input devices. The human operator will be confused by the complex controlling mechanism and thereby the he will constantly think of what inputs should be entered rather than directly manipulating the robot arm, which reflects the difficulty of using such devices.

Our target was to design a telerobotic system in such a way that the human operator can remotely control the motion of the remote arm in natural and easy way and as simple as possible. This target is implemented by using good design of master arm, good mapping schemes, and by building good client-server system. The design of the master arm helps the human operator to forget about the complex controlling

mechanism and to perform any complex motion with it. The development of the mapping schemes allows the human operator to lose the awareness that he is manipulating the master arm and see himself rather manipulating the robot arm directly. The operator can interacts naturally and effortlessly with the robot arm, even with no skills. Our client-server system designed to keep the system reliable and to provides good environment for data transfer. The issues of network reliability, speed of data transfer, amount of data transfered, and informative and simple user interface are all taken into account when we designed our client-server system, which improve the teleoperation system performance.

Chapter 6

EXPERIMENTS AND PERFORMANCE ANALYSIS

This chapter provides details on the results of the telerobotic system and the vision system. In Section 6.1 a detailed description on the vision system performance is given. In Section 6.2 results of the telerobotic system performance are presented.

6.1 Vision System Performance

The vision system performance based on single camera will be discussed in Sub-Section 6.1.1. The issue of the 3D computations and the performance of the vision system to compute the 3D coordinates will be discussed in Sub-Section 6.1.2.

6.1.1 Single Camera Performance

We will concentrate here on the performance of the proposed methods and techniques that are used for color detection and center of gravity evaluation. The following experiments were conducted to test the performance of the designed vision system, all the experiments were conducted by using Sony video cameras

Static Error Measurements

One red ball in a fixed position about 1.5 meter away from the camera is used for this experiment. The target is to measure the relative error in the evaluation of the center of gravity of the non-moving ball. The relative error is measured relative to the ball diameter, where we use the camera zoom to get different ball diameters, from 5 pixels to 65 pixels. Figure 6.1 and Figure 6.2 show the results of this experiment.

Dynamic Error Measurements

For the dynamic error measurements we use the robot arm to hold the ball and to move over a straight line trajectory in one experiment, and in a circular trajectory in the second experiment. The vision system is running during the robot arm motion and tracking the moving ball, at the same time it measures and records the estimated center of the ball continuously. The ball diameter is fixed and is approximately 15 pixels. The number of points that the vision system can record is found to be

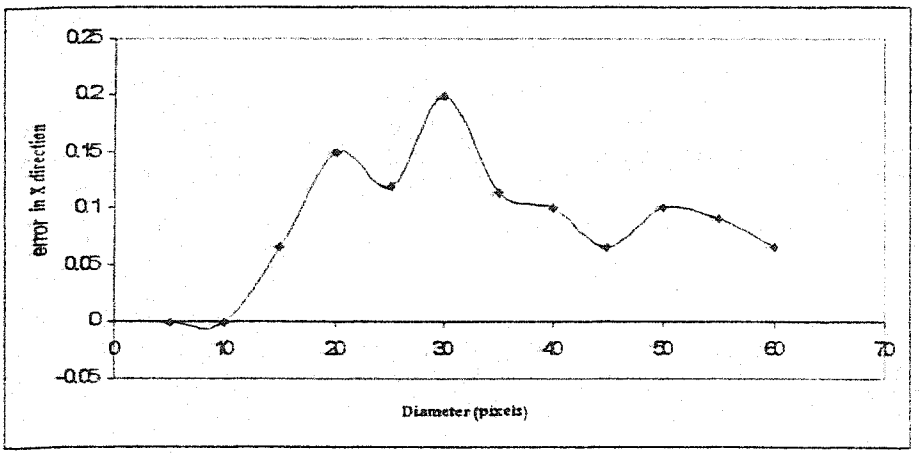

Figure 6.1: Relative error in X direction

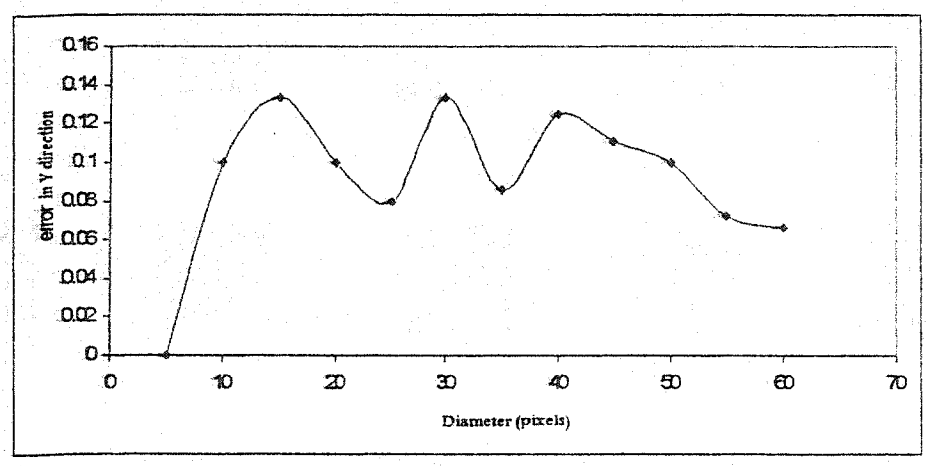

Figure 6.2: Relative error in Y direction

approximately 10 points per second, which represents the frame rate of the image processed by personal computer. The reason for the slow frame rate, given that we use 350MHZ personal computer with 128MB RAM is the amount of computations that are performed by the computer to calculate the center of the ball and the image input overhead.

Figure 6.3 shows the resulted trajectory for the straight line motion implemented at a robot speed of 50 cm/second, while the Figure 6.4 shows the straight line trajectory implemented at a robot speed of 70 cm/second. In Figure 6.4, there are some points which are wrongly detected as red points inside the ball, which means that the vision system will give wrong result for the center of the ball position. Therefore, it is recommend not to move the object at speed higher than this speed. The maximum error for evaluating the center of the ball at speed of 50cm/second was found to be less than 6 pixels and the average error was less than 2 pixels, which are less than the ball diameter and indicate that all the points that are detected are inside the ball but deviate by 2 pixels from the exact center of that ball. In the second experiment the speed of the robot arm is 70 cm/second. The maximum error found to be approximately 58 pixels and the average error was 11.48 pixels. The points that are more than 8 pixels away from the center of the ball are not inside the ball and considered to be error detected points. Therefore, the detection error ratio in this case can be evaluated as the number of error detected points divided by the total number of points which is found to be $16/100 = 0.16\%$, while in the first

case all the points are detected properly and therefore the error in the detection is equal to zero.

In the circular motion tracking, the robot is programmed to move in a circular trajectory at constant speed of 50 cm/second. Our vision system proved that it can track the moving object at that speed without any error in detection as we mentioned and explained in Figure 6.3. Single trajectory circular motion is shown in Figure 6.5, while the multi circle trajectory is shown in Figure 6.6. The results of tracking objects in the circular trajectory are approximately similar to those in the linear trajectory.

The analysis in the dynamic and static experiments show that the vision system is good for tracking and evaluation of the ball center. However, there are some limitations to the system, like the moving speed and the light conditions. For more details about the vision system limitations you may refer to Chapter 4. In the next sub-section we will examine the performance of the vision system when it is used for tracking and computing the 3D information of the vision based master arm.

6.1.2 Three Dimensional Computations Performance

In this section, stereo vision system is used to extract the 3D coordinates of the vision-based master arm using Kuno approach [28, 29], for more details about this method you may refer to section 4.4. The coordinates of the four balls that construct the master arm are detected and measured using two uncalibrated Sony video

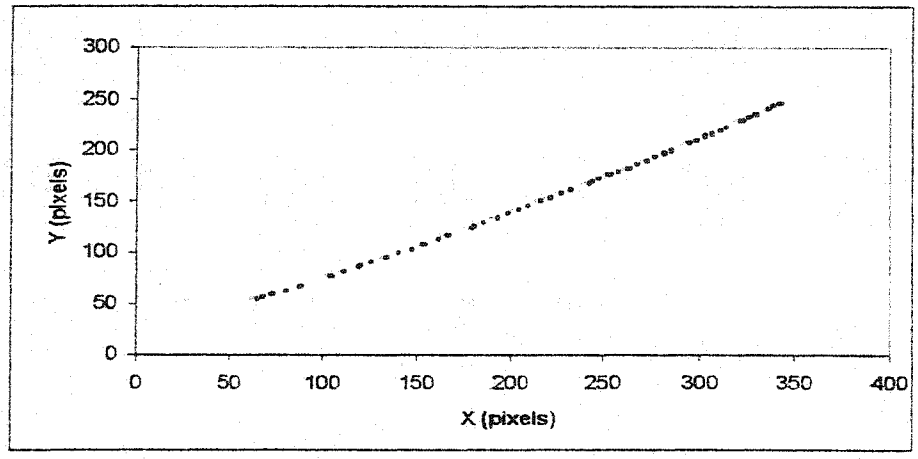

Figure 6.3: Straight line trajectory at moderate speed

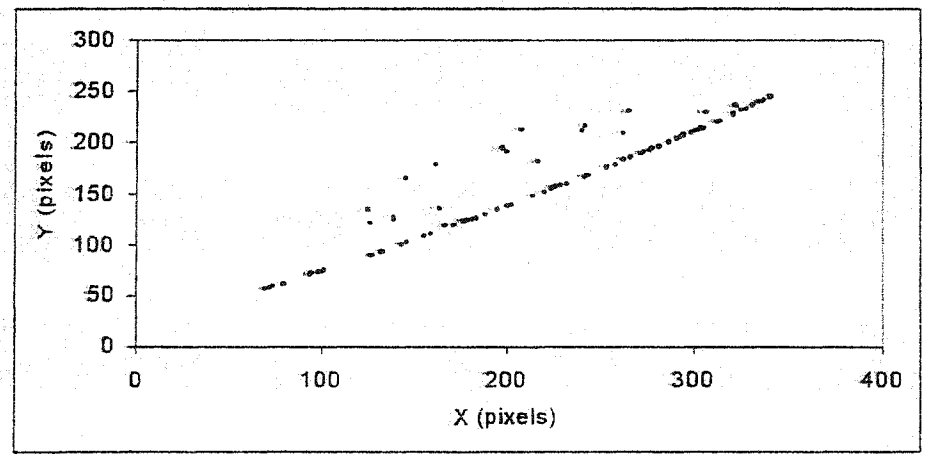

Figure 6.4: Straight line trajectory at high speed

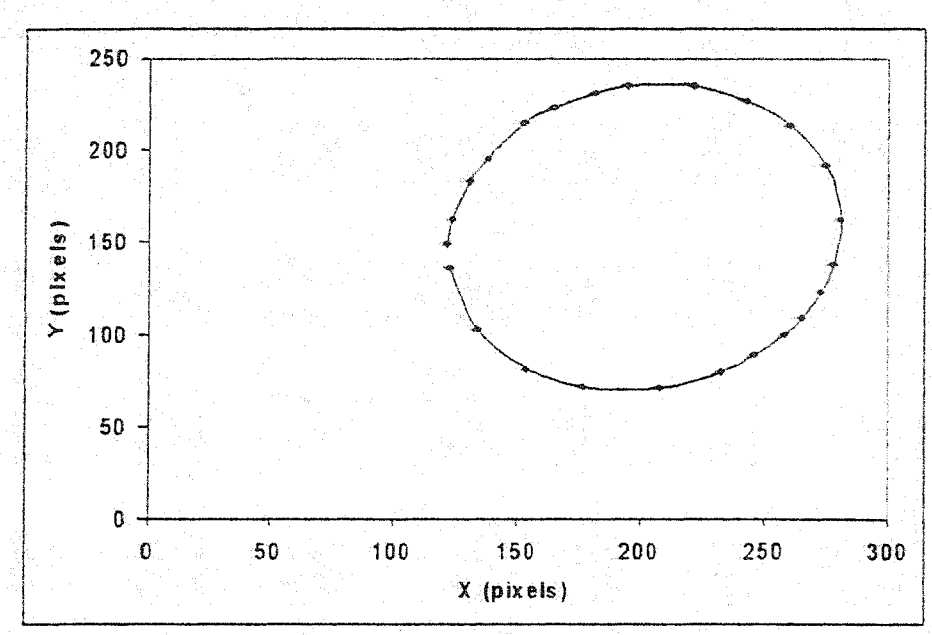

Figure 6.5: Single Circular trajectory

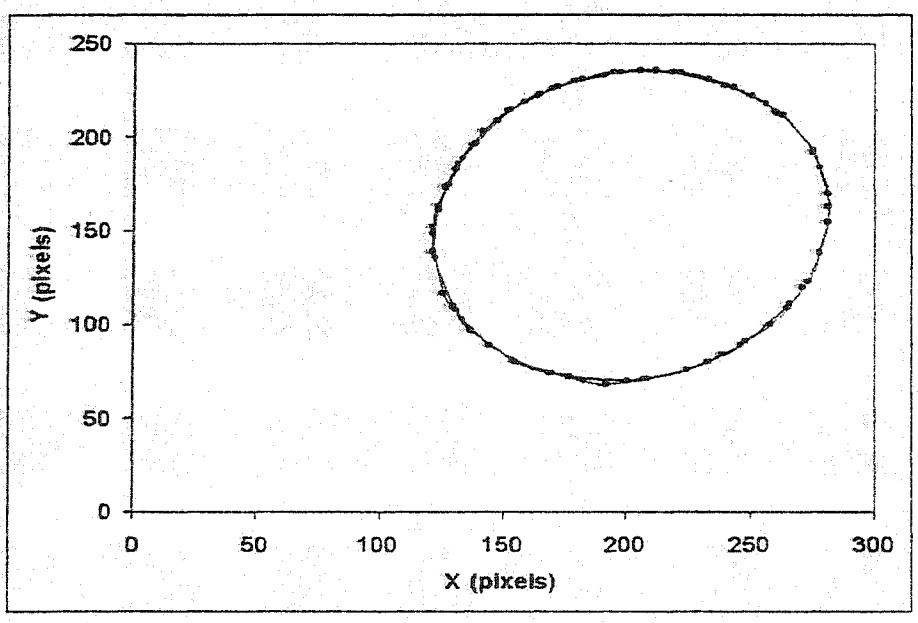

Figure 6.6: Multi Circular trajectory

cameras. The setup for this experiment is shown in Figure 4.13. The video cameras are connected to two different computers, one is used as a client, and the other one as a server, and the vision program is installed on both of them. The computers are connected to each other using serial communication link via RS-232. The client computer receives the image from one video camera and evaluates the coordinates of the four balls in the 2D space, then it sends them to the server computer. The server computer also evaluates the coordinates of the four balls from another view in the 2D space and uses the information that is received from the client to compute the 3D coordinates of the master arm as explained in section 4.4. We use the robot arm to hold the vision-based master arm and move it over a straight line trajectory. In the second experiment the robot moves in a circular trajectory with a constant speed. In both experiments the vision system is running, tracking the moving balls, recording their estimated center and compute the 3-dimensional coordinates for the master arm continuously. The results for the linear trajectory are shown in Figure 6.7, for single direction motion, and in Figure 6.8 for multi motions. The high amount of computation performed at both computers slow down the image processing and consequently the frame rate decreases to less than 5 frames per second. Hence the experiments are conducted at low speed of robot arm motion of 20 cm/second. The maximum error found to be 10 pixels in X direction, 2 pixels in Y direction and 4 pixels in Z direction.

In the second experiment, the robot is programmed to move in a circular tra-

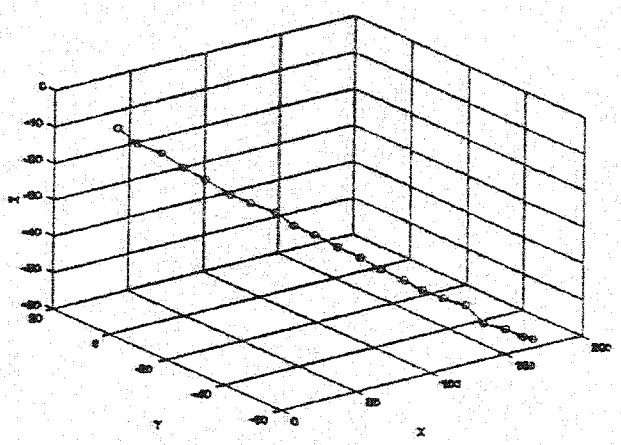

Figure 6.7: Single line trajectory in the 3D space

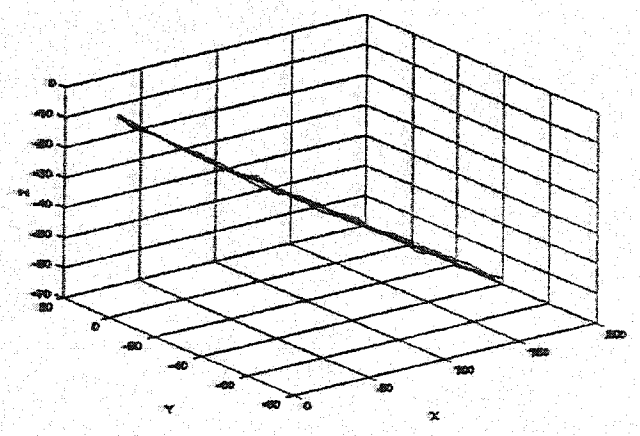

Figure 6.8: Multi line trajectory in the 3D space

jectory and at a fixed speed. The results for single circular trajectory are shown in Figure 6.9 and for multi circular trajectory are shown in Figure 6.10. The maximum error of the measurements found to be 17 pixels in X direction, 4 pixels in Y direction and 9 pixels in Z direction.

The experimental results indicate that the computed position data by the vision system may contain small errors. However, The actual operation experiments show that these errors do not disturb the system performance, and the small deformation in the coordinate system does not affect the operation. The weakness of this system is the slow processing speed, that is we can not move the object fast. Thus, if we use faster computers and faster communication link between them we can get better results.

6.2 Telerobotic System Performance

The telerobotic system consists of the master arm, the client computer, the server computer, the robot arm, the video cameras and the communication link (LAN). when the operator moves the master arm, data-packet will be sent to the server through the network. The server will receive the motion data from the client and will use them to compute the inverse geometric model of the robot, then the server will send motion request to the robot. When the robot finishes the motion, it will send the current position data and a ready signal to the server. Once the server

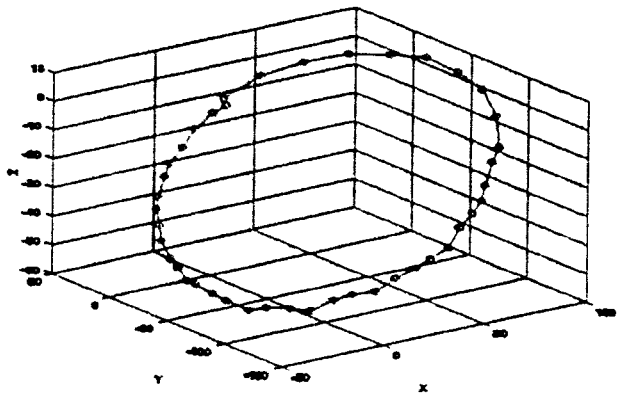

Figure 6.9: Single circle trajectory in the 3D space

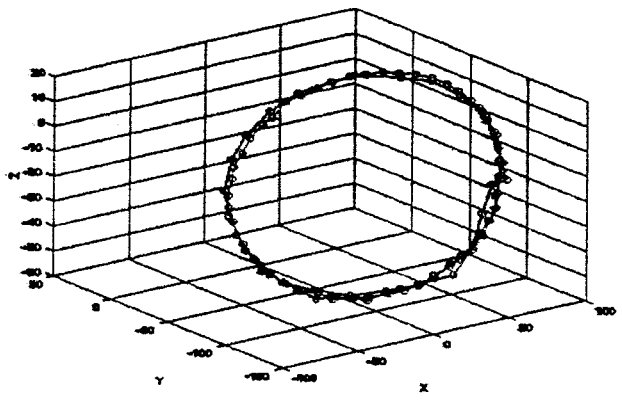

Figure 6.10: Multi circle trajectory in the 3D space

computer receives the ready signal from the robot, it will compute the current direct geometric model of the robot and will send a ready (RDY) signal to the client. The client will respond by acknowledgment (ACK) signal and will send data-packet if any. This operation will continue until the operator stops the motion.

We used the parameters of the telerobotic system to measure and to analyze the performance of each part of the system. The parameters that are measured are:

1. The client data-packet protocol time:
   That is the time taken to send the data from the client PC to the server PC.
2. The robot inverse geometric computation time
   The time that are consumed by the server computer to compute the inverse geometric model of the robot arm.
3. The robot motion and communication time:
   The time taken by the robot to perform the requested motion and to return ready signal to the server.
4. The robot direct geometric computation time:
   That is the time used by the server computer to compute the direct geometric model of the robot arm.
5. C-S round-trip delay: (RDY-ACK) delay:
   This is the round-trip time delay of the LAN.
6. Server-Robot latency measured by the client:
   This is the time measured by the client from the last time it sends data to the server until it receives request from the server to send new data.
7. Client latency measured by the server:
   That is the time measured by the server from the time it sends request to the client to send new data until it receives them completely.

The experiments that are conducted show the performance of the telerobotic system under different conditions. The results of these experiments are given below:

6.2.1 Experiment1: Teleoperation without Video Cameras

The telerobotic system should have video feedback signal from the robot side, but we use this setup to see the effect of the video streams over the network. Figure 6.11 shows the time that is taken by each component during motion request. The contribution of each value to the total is shown in Figure 6.12. The server-robot latency measured by the client and the client latency measured by the server is shown in Figure 6.13. The contribution of the client latency compared to the server latency is shown in Figure 6.14.

Small stops between the motion request is introduced to see their effect on the system, Figure 6.15 and Figure 6.16 show the results of these stops and their effect on the system.

6.2.2 Experiment2: Teleoperation with Two Video Cameras

The results of this experiment show the effect of the video streams over the LAN. Figure 6.17 shows the results of the experiment for the individual parts, and Figure 6.18 shows the server-robot latency measured by the client and the client latency

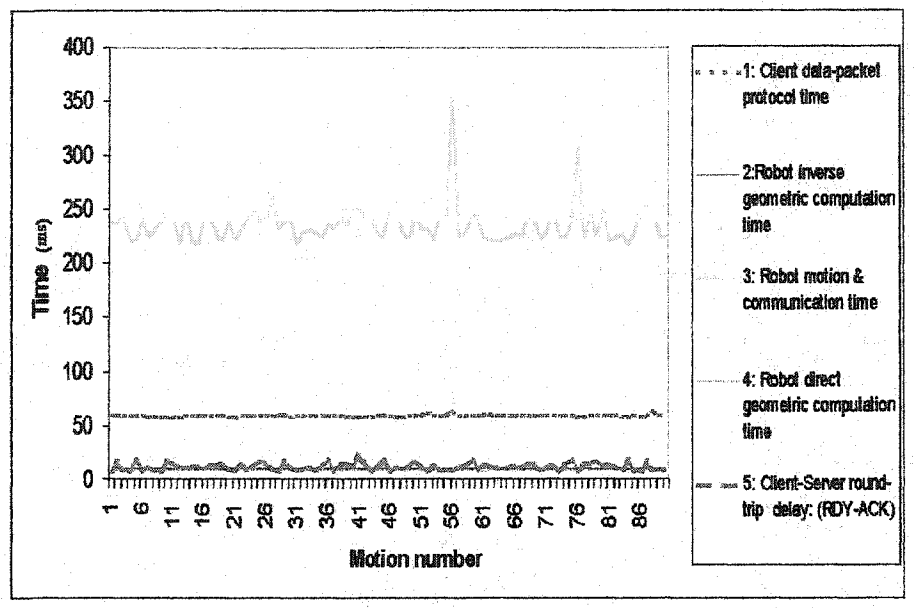

Figure 6.11: Timing in the telerobotic system

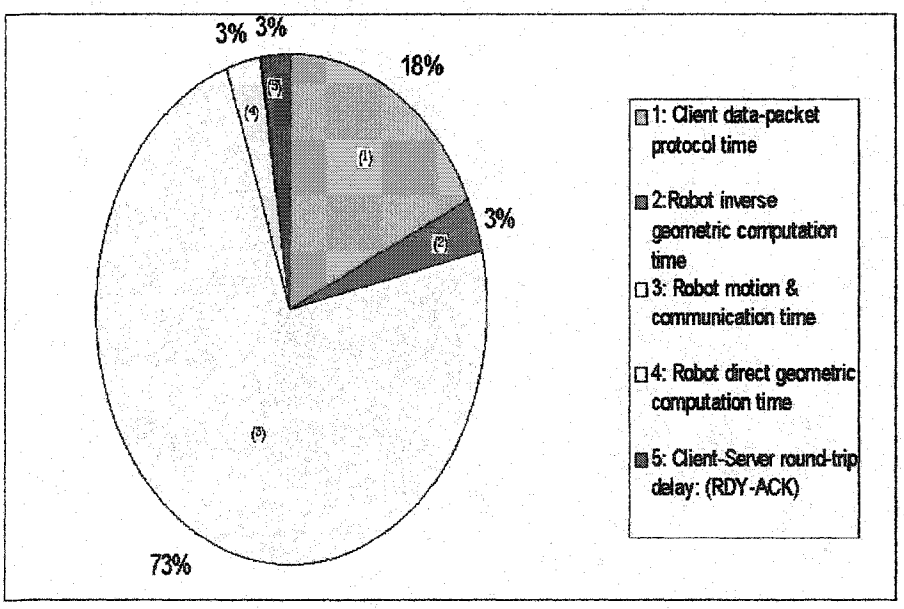

Figure 6.12: Contribution of each value in the system

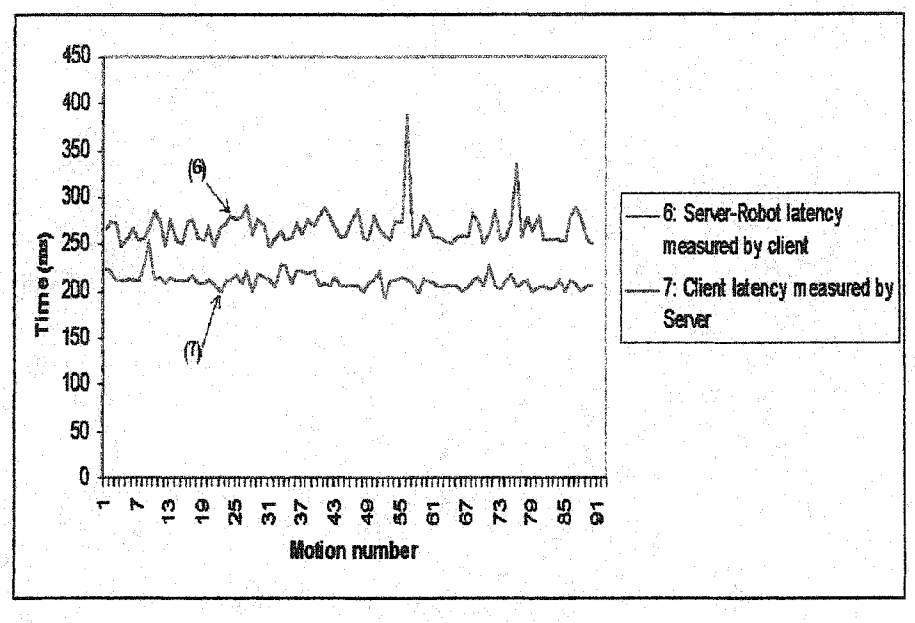

Figure 6.13: Latency measured by the client and the server

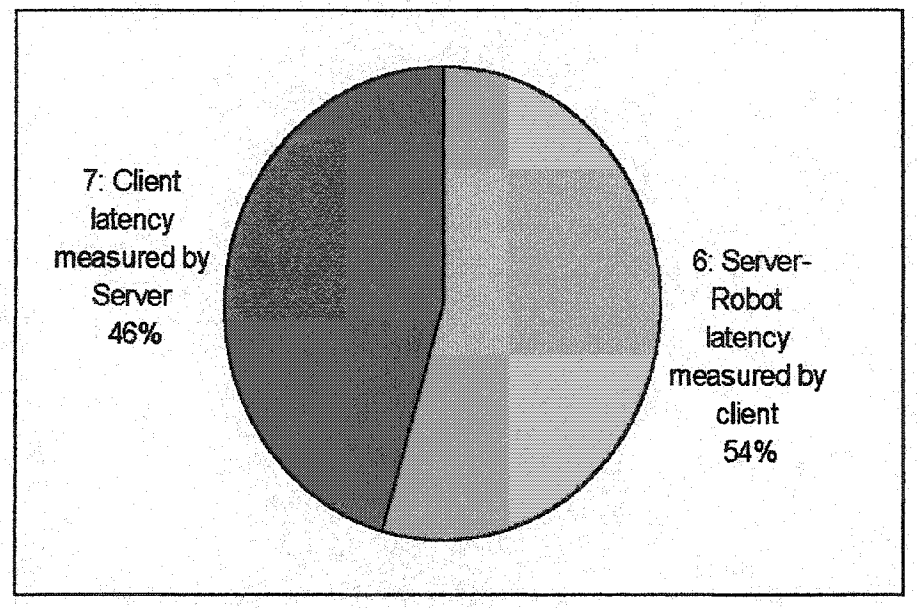

Figure 6.14: Contribution of the client and server as complete system

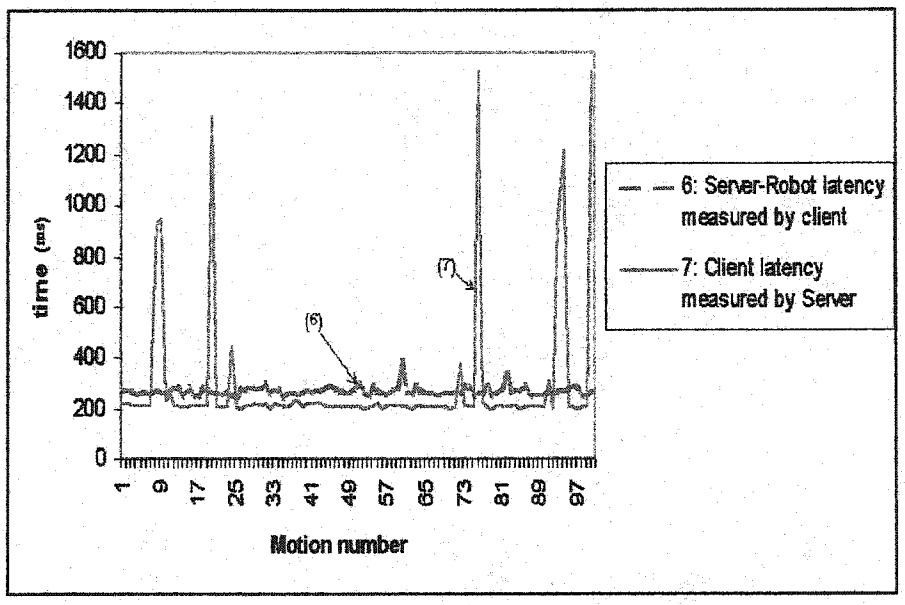

Figure 6.15: The effect of the stops between motions on the client

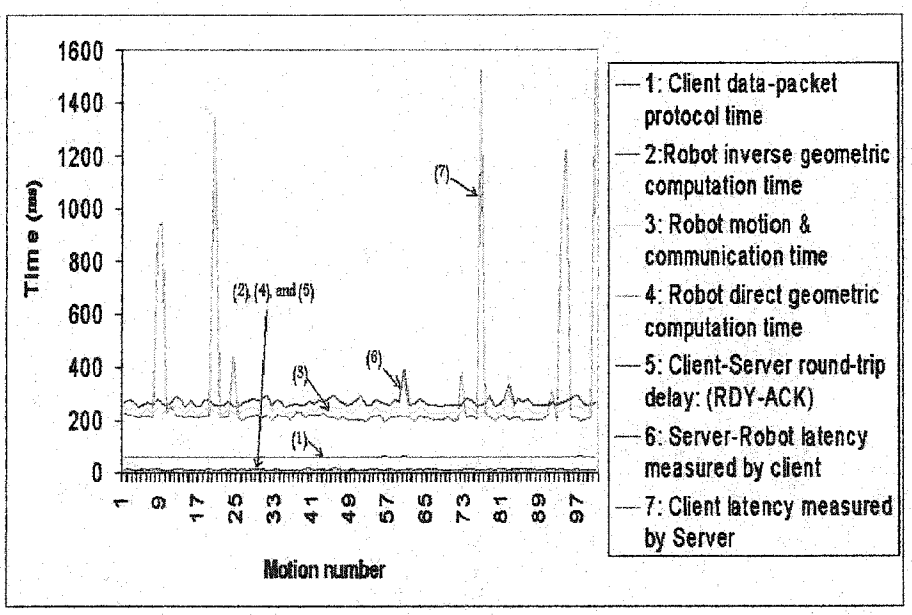

Figure 6.16: The effect of the stops between motions on the whole system

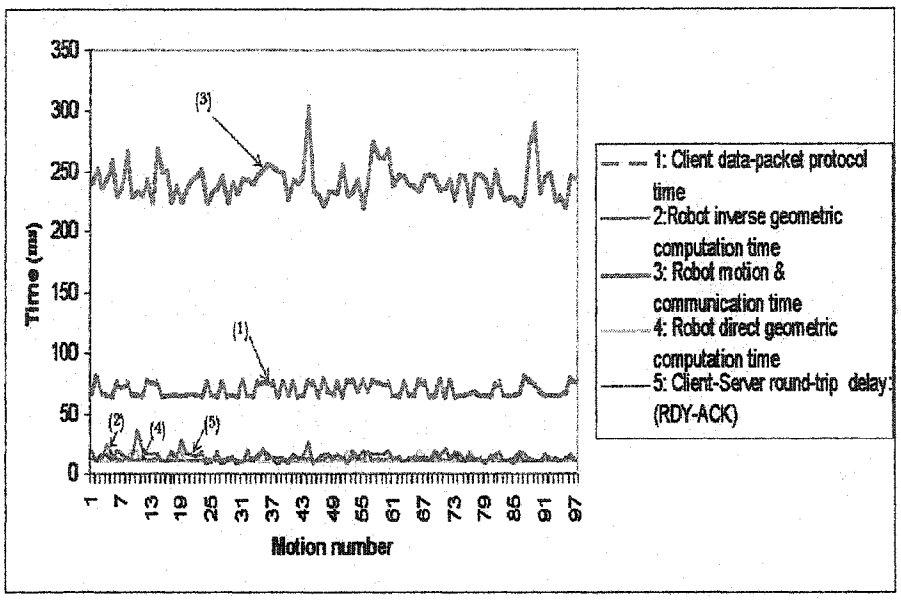

Figure 6.17: Timing in the telerobotic system with video streams feedback

measured by the server under this condition.

Comparison between the average timing of the two experiments is shown in Figure 6.19.

The use of video cameras as shown above did not affect the whole system performance, because of the high bandwidth of the network. The results show that the main contribution is used by the robot motion, smaller amount of time is used to send the data from the client PC to the server PC. The other values are small compared to these two values and should not affect the performance of the telerobotic system.

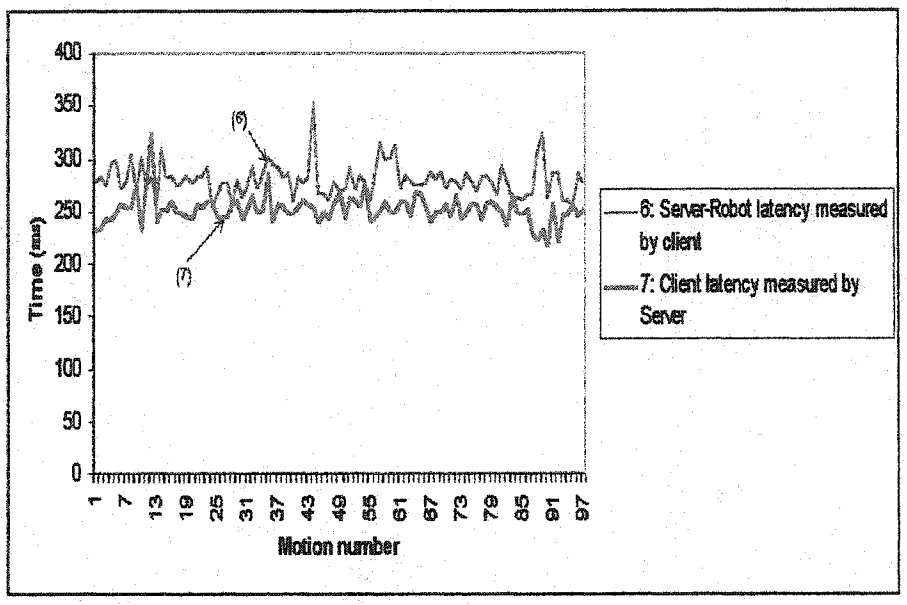

Figure 6.18: Latency measured by the client and the server

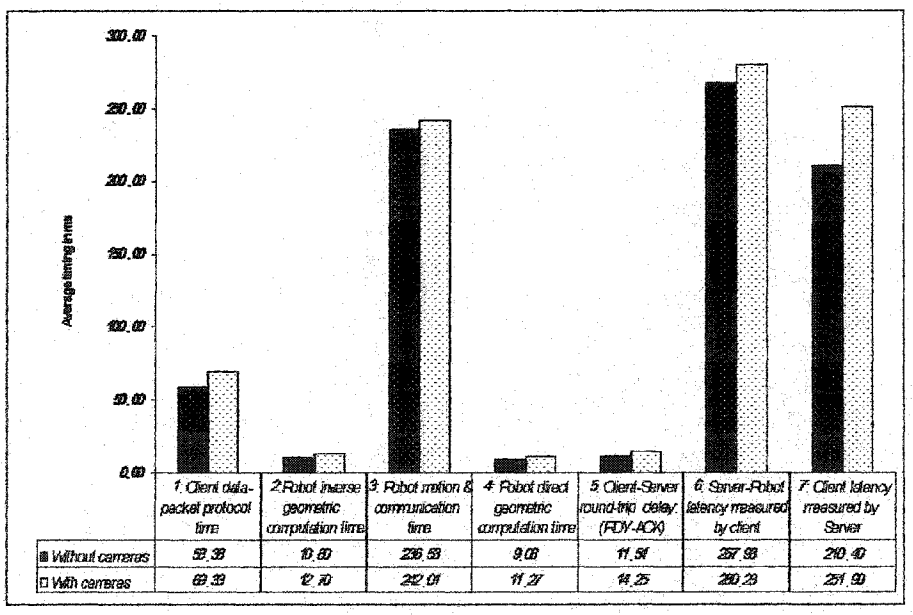

Figure 6.19: Comparison between the average timing

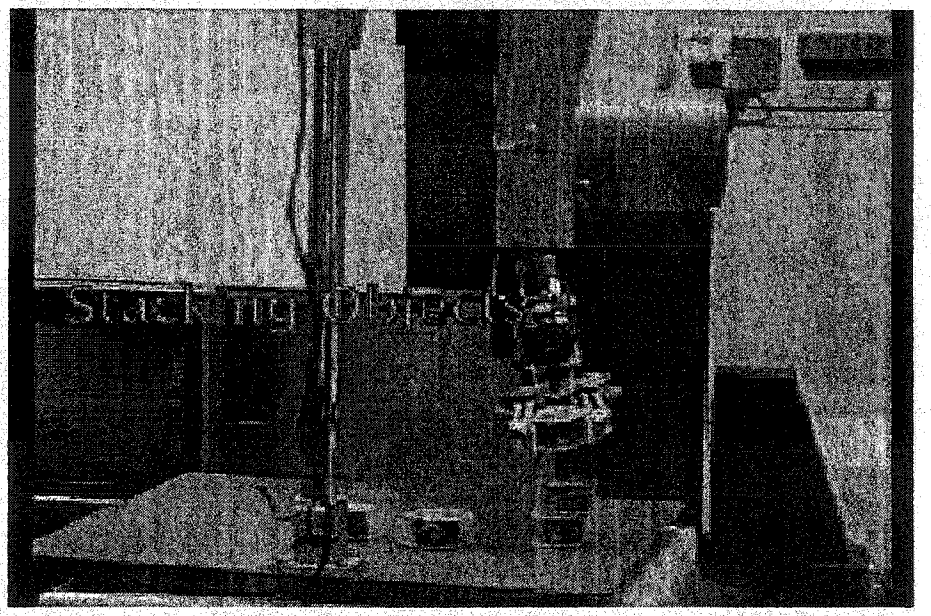

Figure 6.20: Stacking objects

6.3 Actual Experiments

The standard performance measures for teleoperation are essentially called time to complete given tasks and arbitrarily devised scores for how well they are completed (accuracy and error). We experimented our system and we got good result in term of time to complete tasks and the accuracy of performance. The tasks we used in our experiments include manipulation tasks such as positioning or stacking blocks as shown in Figure 6.20 or writing words like that shown in Figure 6.21. These are tasks that involve grasping and ungrasping, free movement as well as movement near obstacles, and fine adjustments. At the moment there are essentially no accepted standards for asserting that one telemanipulator system is better or worse than some other.

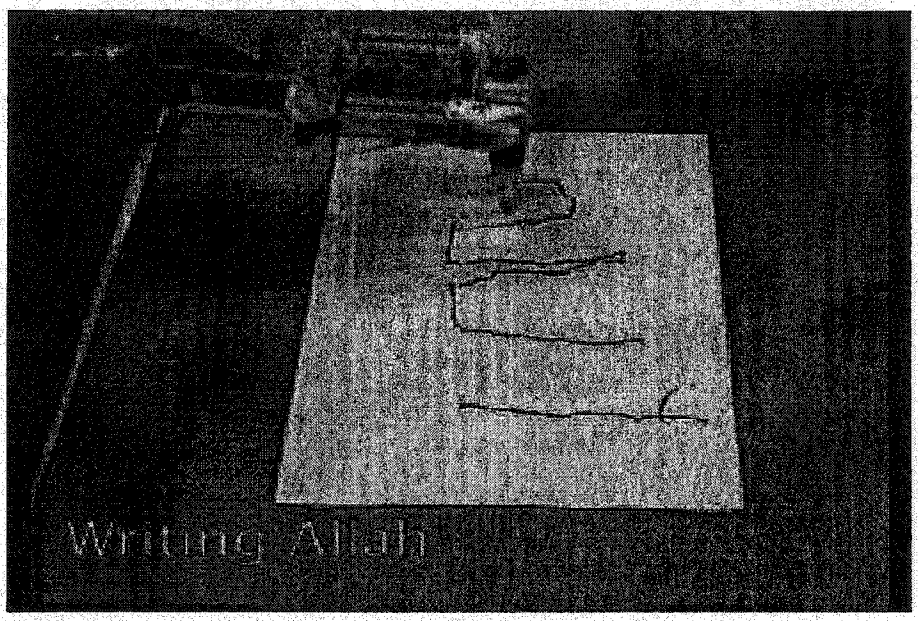

Figure 6.21: Writing Allah

Chapter 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

Transferring human manipulative capability over a local area network to remotely operate manipulator or vehicle with direct or supervisory control was the main target for this research. The term of telerobotic system was introduced to represent the system that can do this work. For the design of the telerobotic system, a number of problems were solved such as defining a general network interface for the master and the slave workstations, implementing network connectivity and transfer of control, handling the problem of the difference in the geometric designs between the master and the slave arms, and providing visual information.

A client-server system is developed as the general interface for the master and slave workstations. It also provides the connectivity and transfer of data and control between the different parts of the telerobotic system. Some issues like the amount of data to be transfered, the user interface, and the reliability of the network are all taken into account when the client-server was designed.

Two mapping schemes were presented in this thesis to solve the problem of the difference in the geometric designs between the master arm and the slave arm (PUMA-560). A comparison between both mapping schemes has been made and showed that the second mapping scheme is better than the first one.

The telerobotic system was tested experimentally and it gave good results in term of time to complete given tasks and accuracy of this work.

We also developed vision-based master arm, which may replace the mechanical master arm. In the vision-based master arm we used uncalibrated stereo vision system to extract the three dimensional information of the moving master arm from the multiple views. The vision-based master arm which was developed in this research, showed that the use of this master arm instead of the classical one is possible, especially upon the availability of fast computers at low cost.

7.2 Recommendations

Following are some suggested extensions to this research work:

1. Replacing the existing master arm by a force-reflection (FR) master arm will allow the operator to feel the reflected force feedback and can dramatically improve the task execution performance.

2. Virtual-reality of slave scenes, can be implemented by conveying stereo vision information and connecting this activity to the eye of the operator using head mounted displays (HMDs) in order to create a proper visual environment for telepresence or teleexistence.

3. The robot arm can be equipped with some type of sensors, according to the work needs. These sensors will allow automatic execution of functions that are handled with difficulty by the operator and they will allow the robot arm to take some actions during the absence of the operator. This type of operation is required in most of the telerobotics applications.

Bibliography

[1] Bejczy, A.K. "Challenge of human-robot communication in telerobotics". Robot and Human Communication, 1996., 5th IEEE International Workshop on , 1996 Page(s): 1 -8

[2] Bejczy, A.K.; Kim, W.S.; Venema, S.C. "The phantom robot: predictive displays for tyeleoperation with time delay". Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on , 1990 Page(s): 546 -551 vol.1

[3] Arthur J. Critchlow. Introduction to Robotics. Macmillan Publishing Company, 1985

[4] P. G. Bakes, "multi-sensor Based impedance control for task execution". In Proc. Of the 1992 IEEE Intl. Conf. On Robotics and Automation.

[5] Dalton, B.; Taylor, K. "Distributed robotics over the internet". IEEE Robotics and Automation Magazine, Volume: 7, Issue: 2 , June 2000 Page(s): 22 -27

[6] Triggs, B.; Laugier, C. "Automatic camera placement for robot vision tasks". Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on , Volume: 2, 1995 Page(s): 1732 -1737

[7] Tanaka, K.; Nakagawa, E.; Ito, M.; Mizuno, N.; Yamada, T.; Shimizu, E.; Kagayama, K. "An internet based tele-robot environment for a time critical task". Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on , Volume: 5, 1999 Page(s): 1106 -1110

[8] Dony, R.D.; Wesolkowski, S. "Edge detection on color images using RGB vector angles". Electrical and Computer Engineering, 1999 IEEE Canadian Conference on , Volume: 2, 1999 Page(s): 687 -692

[9] Everett, S.E.; Isoda, Y.; Dubey, R.V.; Dumont, C. "Vision-based end-effector alignment assistance for teleoperation". Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on , Volume: 1, 1999 Page(s): 543 -549

[10] Goktas, F.; Smith, J.M.; Bajcsy, R. "Telerobotics over communication networks". Decision and Control, 1997., Proceedings of the 36th IEEE Conference on , Volume: 3, 1997 Page(s): 2399 -2404

[11] Dong, F.F.; Meng, M. "A computer 3D animated graphical interface for control and teleoperation of robot system". Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on , Volume: 1, 1997 Page(s): 778 -783

[12] Leung, G.M.H.; Francis, B.A.; Apkarian, J. "Bilateral controller for teleoperators with time delay via -Synthesis". Robotics and Automation, IEEE Transactions on , Volume: 11, Issue: 1 , Feb. 1995 Page(s): 105 -116

[13] Gil-Whoan Chu; Myung Jin Chung "An optimal image selection from multiple cameras under the limitation of communication capacity". Multisensor Fusion and Integration for Intelligent Systems, 1999. MFI '99. Proceedings. 1999 IEEE/SICE/RSJ International Conference on , 1999 Page(s): 261 -266

[14] Ken Goldberg. " The Robot in the Garden : Telerobotics and Telepistemology in the age of the internet". The MIT Press,2000.

[15] C. Guo, T. J. Tarn, N. Xi and A. K. Bejczy. "Fusion of human and machine intelligence for telerobotics systems." In Proc. Of the 1995 IEEE Intl. Conf. On Robotics and Automation

[16] Friz, H.; Elzer, P.; Dalton, B.; Taylor, K. "Augmented reality in internet telerobotics using multiple monoscopic views". Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on , Volume: 1, 1998 Page(s): 354 -359

[17] Haule, D.D.; Malowany, A.S. "Control scheme for delayed teleoperation tasks". Communications, Computers, and Signal Processing, 1995. Proceedings., IEEE Pacific Rim Conference on , 1995 Page(s): 157 -160

[18] S. Hayati and S. T. Venkataraman. "Design and implementation of a Robot control system with traded and shared control capability". In Proc. Of the 1989 IEEE international Conference on Robotics and automation.

[19] Kakeya, H.; Oyama, K.; Arakawa, Y. "3D display system for reality-enhanced teleoperation". Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on , Volume: 5, 1999 Page(s): 1129 -1134

[20] C.Y. Ho and Jen Sriwattanatamma. "Robot Kinematics". Ablex Publishing Corporation, 1990

[21] Belousov, I.R.; JiaCheng Tan; Clapworthy, G.J. "Teleoperation and JAVA 3D visualization of a robot manipulator over the world wide web". Information Visualization, 1999. Proceedings. 1999 IEEE International Conference on , 1999 Page(s): 543 -548

[22] Jong-Shin Lee; Hyeog-Min Kwon; Dong-Yoon Shin; Jae-Bok Song "Force feedback using sensibility ergonomics theory in teleoperation system". Intelligent Robots and Systems, 1999. IROS '99. Proceedings. 1999 IEEE/RSJ International Conference on , Volume: 1, 1999 Page(s): 597 -602

[23] Kosuge, K.; Murayama, H.; Takeo, K. "Bilateral feedback control of telemanipulators via computer network". Intelligent Robots and Systems '96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on , Volume: 3, 1996 Page(s): 1380 -1385

[24] Kosuge, K.; Murayama, H. "Bilateral feedback control of telemanipulator via computer network in discrete time domain'. Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on , Volume: 3, 1997 Page(s): 2219 -2224

[25] Goldberg, K.; Mascha, M.; Gentner, S.; Rothenberg, N.; Sutter, C.; Wiegley, J. "Desktop teleoperation via the world wide web". Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on , Volume: 1, 1995 Page(s): 654 -659

[26] Ken Taylor, Barney Dalton. "Issues in internet telerobotic". 1999.

[27] Brady, K.; Tzyh-Jong Tarn "Internet-based remote teleoperation ". Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on , Volume: 1, 1998 Page(s): 65 -70

[28] Kuno, Y.; Sakamoto, M.; Sakata, K.; Shirai, Y. "Vision-Based Human Interface with User-Centered Frame". Intelligent Robots and Systems '94. 'Advanced Robotic Systems and the Real World', IROS '94. Proceedings of the IEEE/RSJ/GI International Conference on , Volume: 3, 1994 Page(s): 2023 -2029

[29] Kuno, Y.; Hayashi, K.; Jo, K.H.; Shirai, Y. "Human-Robot Interface Using Uncalibrated Stereo Vision". Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on , Volume: 1, 1995 Page(s): 525 -530

[30] Leleve, A.; Fraisse, P.; Dauchez, P.; Pierrot, F. "Modeling and simulation of robotics tasks teleoperated through the internet". Advanced Intelligent Mechatronics, 1999. Proceedings. 1999 IEEE/ASME International Conference on , 1999 Page(s): 299 -304

[31] Mallem, M.; Colle, E.; Chavand, F. "A multimedia control interface in teleoperation". Systems, Man and Cybernetics, 1993. 'Systems Engineering in the Service of Humans', Conference Proceedings., International Conference on , 1993 Page(s): 126 -131 vol.3

[32] Martin Jagersand. "Image based predictive display for tele-manipulation". Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on , Volume: 1, 1999 Page(s): 550 -556

[33] M. Al-Mouhamed. Introduction to Robotics. 2000.

[34] Nak Young Chong; Ohba, K.; Kotoku, T.; Komoriya, K.; Matsuhira, N.; Tanie, K. "Coordinated rate control of multiple telerobot systems with time delay". Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on , Volume: 5, 1999 Page(s): 1123 -1128

[35] Funabiki, N.; Morishige, K.; Noborio, H. "Sensor-based motion planning of a manipulator to overcome large transmission delays in teleoperation". Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on , Volume: 5, 1999 Page(s): 1117 -1122

[36] Ning Xi; Tarn, T.J. "Action synchronization and control of internet based telerobotic systems". Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on , Volume: 1, 1999 Page(s): 219 -224

[37] Fiorini, P.; Oboe, R. "Internet based telerobotics: problems and approaches". Advanced Robotics, 1997. ICAR '97. Proceedings., 8th International Conference on , 1997 Page(s): 765 -770

[38] Paolucci, F.; Andrenucci, M. "Teleoperation using computer networks: prototype realization and performance analysis". Electrotechnical Conference, 1996. MELECON '96., 8th Mediterranean , Volume: 2, 1996 Page(s): 1156 -1159

[39] Milgram, P.; Rastogi, A.; Grodski, J.J. "Telerobotic control using augmented reality". Robot and Human Communication, 1995. RO-MAN'95 TOKYO, Proceedings. 4th IEEE International Workshop on , 1995 Page(s): 21 -29

[40] Penin, L.F.; Matsumoto, K.; Wakabayashi, S. "Force reflection for time delayed teleoperation of space robots". Robotics and Automation, 2000. Proceedings.

ICRA '00. IEEE International Conference on , Volume: 4, 2000 Page(s): 3120 -3125

[41] Tsang, P.W.M.; Tsang, W.H. "Edge detection on object color". Image Processing, 1996. Proceedings. International Conference on , Volume: 3, 1996 Page(s): 1049 -1052

[42] Inoue, S.; Ojika, T.; Harayama, M.; Kobayashi, T.; Imai, T. " Two dimensional control for 6-DOF hand robot teleoperator ". Robot and Human Communication, 1993. Proceedings., 2nd IEEE International Workshop on , 1993 Page(s): 171 -176

[43] Thomas B. Sheridan. Telerobotics, Automation, and Human supervisory control. The MIT Press, 1992

[44] Shinjiro Kawato and Jun Ohya. "Real-time Detection of Nodding and Headshaking by Directly and Tracking the "Between-Eyes"". 2000.

[45] Slutski, L.; Edan, Y.; Friedman, L. "Remote control means study for telerobotic operations". Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on , Volume: 3, 1998 Page(s): 1609 -1614

[46] Soeda, M.; Warwick, K.; Craddock, R.J.; Furuya, T. "Human-computer cooperative teleoperation with time delay". Control '98. UKACC International Conference on (Conf. Publ. No. 455) , 1998 Page(s): 1444 -1449 vol.2

[47] Tzafestas, S.G.; Tzafestas, C.S. "Virtual reality in telerobotics: the state of the art". Industrial Electronics, 1999. ISIE '99. Proceedings of the IEEE International Symposium on , Volume: 1, 1999 Page(s): 280 -286

[48] Sukhan Lee; Sookwang Ro; Jong-Oh Park; Chong-Won Lee "Optimal 3D viewing with adaptive stereo displays: a case of tilted camera configuration'. Advanced Robotics, 1997. ICAR '97. Proceedings., 8th International Conference on , 1997 Page(s): 839 -844

[49] Fitzpatrick, T. "Live remote control of a robot via internet". IEEE Robotics and Automation Magazine , Volume: 6 Issue: 3 , Sept. 1999 Page(s): 7 -8

[50] Suzuki, T.; Fujii, T.; Yokota, K.; Asama, H.; Kaetsu, H.; Endo, I. "Teleoperation of multiple robots through the internet". Robot and Human Communication, 1996., 5th IEEE International Workshop on , 1996 Page(s): 545 -546

[51] Wen-Hong Zhu; Salcudean, S.E. "Teleoperation with adaptive motion/force control". Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on , Volume: 1, 1999 Page(s): 231 -237

[52] Stark, L.W.; Kim, W.S.; Tendick, F. "Cooperative control in telerobotics". Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on , 1988 Page(s): 593 -595 vol.1

[53] Kim, W.S.; Bejczy, A.K. "Demomstration of a high fidelity predictive/preview display technique for telerobotic servicing in space". Robotics and Automation, IEEE Transactions on , Volume: 9 Issue: 5 , Oct. 1993 Page(s): 698 -702

[54] Yeuk Fai Ho; Masuda, H.; Oda, H.; Stark, L.W. "Distributed control for teleoperation". Mechatronics, IEEE/ASME Transactions on , Volume: 5 Issue: 2 , June 2000 Page(s): 100 -109

Vita

- Abdulkhaliq Jraib Alharthi
- Born in Abu-Dhabi, UAE
- Received Bachelor's degree in Electrical Engineering from United Arab Emirates University, UAE, 1994
- Worked as Control Engineer at Abu-Dhabi National Oil Company (ADNOC) since 1994.
- Joined Systems Engineering Dept. in Spring 1999.
- Completed Master's degree requirements at King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia in Spring 2002.